THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Development and Implementation of Robust Reconstruction Methods for Small-Angle X-ray Scattering Tensor Tomography

Leonard C. Nielsen

Department of Physics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2024

Development and Implementation
of Robust Reconstruction Methods
for Small-Angle X-ray Scattering
Tensor Tomography
Leonard Nielsen

isbn 978-91-8103-055-6
Doktorsavhandlingar vid Chalmers tekniska högskola. Ny serie nr 5513
issn 0346-718X

Department of Physics
Chalmers University of Technology
se-412 96 Göteborg, Sweden
Telephone +46 (0)31 772 10 00

Cover: 3D render of tensor field reconstucted via tensor tomography, from a sample of trabecular bone.

Chalmers digitaltryck
Göteborg, Sweden 2024

Development and Implementation of Robust Reconstruction Methods for Small-Angle X-ray Scattering Tensor Tomography

Leonard C. Nielsen
Department of Physics
Chalmers University of Technology

Abstract

This thesis details the development of an improved theoretical framework, computational methods, and the implementation of scientific software for small-angle x-ray scattering tensor tomography. It includes experimental and simulation work to evaluate the robustness of the method under constraints on data acquisition and to validate implications of the theory.

Small-angle x-ray scattering tensor tomography is a promising method for the study of millimeter-sized samples with anisotropic nanostructures at scales ranging from a few to several hundred nanometers. This is accomplished by reconstructing the reciprocal space map of the sample in a volume-resolved manner. A thorough account is given of the mathematical model and the algorithms used to carry out reconstructions from experimental data. The performance of these algorithms is investigated through correlation studies on simulated data, demonstrating that they are robust and perform well compared to alternative approaches. The thesis details the software package Mumott, the structure and development of which is informed by the advances within this project. In the case of the experimental correlation study, the robustness of the tensor tomographic reconstruction methods for a typical partial data set is evaluated by comparison to the reconstruction obtained with a full data set. The full data set was acquired by remounting the sample during acquisition. This analysis shows that while the incompleteness of data has an appreciable adverse effect on the parts of the reconstruction, many important quantities, such as the principal orientation, mean intensity and relative anisotropy, are relatively robust. Finally, the thesis includes one application study of the structure of the chameleon tongue, where small-angle x-ray scattering tensor tomography is used to complement other imaging methods in a way that leverages the advances in robustness and the study of complex textures detailed in the thesis.

Keywords: small-angle scattering, tensor tomography, visualization, nanostructure, diffraction, hierarchical materials, integral geometry, optimization

LIST OF APPENDED PAPERS

This thesis is partly based on the author's licentiate thesis (L. C. Nielsen, *Theoretical and computational advances in small-angle x-ray scattering tensor tomography* (2022)). It consists of 8 chapters and the following papers:

I **Small-angle scattering tensor tomography algorithm for robust reconstruction of complex textures**
L. C. Nielsen, P. Erhart, M. Guizar-Sicairos, and M. Liebi
Acta Crystallographica **A79**, 515-526 (2023)

II **Investigating the missing wedge problem in small-angle x-ray scattering tensor tomography across real and reciprocal space**
L. C. Nielsen, T. Tänzer, I. Rodriguez-Fernandez, P. Erhart, and M. Liebi
Submitted to Journal of Synchrotron Radiation

III **Structure of the Chameleon Tongue: Enabling one of the fastest movements in biology**
J. Avaro*, **L. C. Nielsen***, C. Appel, D. Athansiadou, A. Parilli, F. Orellana, L. Davis, U. Hetzel, and M. Liebi
**Both authors contributed equally.*
In manuscript

IV **Mumott – a Python package for tensor tomography**
L. C. Nielsen, M. Carlsen, S. Wang, A. Baroni, T. Tänzer, M. Liebi, and P. Erhart
In manuscript

The author's contribution to the papers:

I I developed the improvements to the reconstruction algorithm, wrote the software, developed the formalism, carried out the computations, analyzed the results, developed and implemented the simulation algorithm used, created the figures, and wrote the majority of the text.

II I authored the experimental proposal, participated remotely in the experiment, developed the methods used to analyze the data, carried out the reconstruction and analysis, wrote the code used to carry out the analysis, interpreted the results, wrote the majority of the text, and created the figures.

III I co-wrote the manuscript together with the other authors. I carried out the SASTT reconstruction and analysis, and developed the methods for extracting multiple orientations. I created the SASTT figures, wrote the corresponding method part and contributed to the writing of the overall manuscript.

IV I wrote the manuscript together with the other co-authors. I am the principal author of the software Mumott that the paper details, and implemented most of the existing code. I developed the formalism which is used to approach the problem of tensor tomography within the software.

PUBLICATIONS NOT INCLUDED IN THIS THESIS

The following publications are outside the scope of this thesis:

Micro- and nanostructure specific X-ray tomography reveals less matrix formation and altered collagen organization following reduced loading during Achilles tendon healing
Silva Barreto, I., Pierantoni, M., **Nielsen, L. C.**, Hammerman, M., Diaz, A., Novak, V., Eliasson, P., Liebi, M., Isaksson, H.
Acta Biomaterialia **174**, 245-257 (2024)

SAXS imaging reveals optimized osseointegration properties of bioengineered oriented 3D-PLGA/aCaP scaffolds in a critical size bone defect model
Casanova, E. A., Rodriguez-Palomo, A., Stähli, L., Arnke, K., Gröninger, O., Generali, M., Neldner, Y., Tiziani, S., Dominguez, A. P., Guizar-Sicairos, M., Gao, Z., Appel, C., **Nielsen, L. C.**, Georgiadis, M., Weber, F. E., Stark, W., Pape, H. C., Cinelli, P., Liebi, M.
Biomaterials **294**, (2023)

Unveiling breast cancer metastasis through an advanced X-ray imaging approach
Conceição, A. L. C., Müller, V., Burandt, E.-C., Mohme, M., **Nielsen, L. C.**, Liebi, M., Haas, S.
Scientific Reports **14(1)**, (2024)

Lack of embryonic skeletal muscle in mice leads to abnormal mineral deposition and growth
Silva Barreto, I., Liebi, M., Le Cann, S., Ahmed, S., **Nielsen, L. C.**, Grünewald, T.A., Dejea, H., Lutz-Bueno, V., Nowlan, N., Isaksson, H.
In review

Nanostructural evolution during carious and demineralisation process of human dentine using SAXS tensor tomography
Rabnawaz, T., Leung, N., **Nielsen, L. C.**, Robert, H. A., Snow, T., Shelton, R. M., Landini, G., Smith, A. J., Terrill, N. J., Liebi, M., Sui, T.
In review

Acknowledgments

First and foremost, I would like to thank my supervisor, Marianne Liebi, and my co-supervisor, Paul Erhart, for giving me the opportunity to carry out this work, as well as guiding and supporting me through it. I also want to thank Aleksandar Matic, who in his capacity as examiner and division head supported me throughout my nearly five years as a Ph. D. student. I am also grateful towards all of my collaborators, co-authors and colleagues who made it possible for me to pursue, present, author and publish research throughout these years, including everybody in the Liebi research group. I also acknowledge the Swedish Research Council (Vetenskapsrådet) and the European Research Council for funding this research; the Paul Scherrer Institut of Villigen, Switzerland, and the cSAXS beamline at the Swiss Light Source for providing beamtime; and Chalmers e-Commons for providing some of the computational resources necessary to carry out this work.

In addition, I particularly want to extend my gratitude to Christian Appel, who both during and after his affiliation with the Liebi group has always gone above and beyond in giving support with many matters, from theory to technical issues, software development, and career discussions. Moreover, I want to thank all of those who have tested, used, and given feedback on the Mumott software. In particular I want to highlight those who used it already in its pre-alpha stages, especially Adrian Rodriguez Palomo, Isabella Silva Barreto, and Tayyaba Rabnawaz.

I thank the many people at the Division of Materials Physics at Chalmers who assisted me with various minor practical things, such as restarting my computer, during the large portion of my Ph. D. for which I have worked remotely. In the same vein, I would like to thank everybody who in various contexts and ways have helped or offered to help me when my health has limited me. Without you, this would not have been possible for me.

A few other individuals who I have come to know over the course of this work come to mind as bearing mention.

Yuqing Ge, for her friendship, and for the many times that she has listened to me, talked to me, encouraged me, and offered her perspective. The value of this support is difficult for me to overstate.

Mirna Alhanash, for always offering friendly, interesting and engaging conversation during my occasional presence at the office. Sometimes small things leave a lasting impression.

And finally, I must thank my wife, Hezhe. I could not possibly have had the persistence to finish this work without her love, help and support.

List of abbreviations

FBP filtered back-projection. 45

GPU graphics processing unit. 28

IR iterative reconstruction. 62, 63, 65, 66

IRLS iteratively reweighted least squares. 39, 42, 45

IRN iteratively reweighted norm. 39, 42, 43

LERP linear interpolation. 25

MITRA modular iterative tomographic reconstruction algorithm. 58, 67

MOTR momentum total variation reconstruction. 43, 58

RADTT robust and denoised tensor tomography. 58

RBF radiaul basis function. 22

RSM reciprocal space map. 1, 2, 4–9, 11–13, 20, 33, 41–43, 46, 47, 57, 63–71, 73, 74, 76–78, 80, 82, 85–89, 91, 92

SAXS small-angle x-ray scattering. 1–6, 10, 51, 68, 80, 83, 84, 89, 92

SAXSTT small-angle x-ray scattering tensor tomography. 2–4, 13, 14, 17, 19, 20, 35, 36, 46, 52, 59–62, 66, 70, 73, 81–84, 91, 92

SEM scanning electron microscopy. 1, 2

SIGTT spherical integral geometric tensor tomography. 58, 60, 62, 63, 66

SIRT simultaneous iterative reconstruction technique. 37, 38, 42, 43, 58

SLERP spherical linear interpolation. 25

SNR signal-to-noise ratio. 65–67

sSAXS scanning SAXS. 1, 2

WAXS wide-angle x-ray scattering. 5

WAXSTT wide-angle x-ray scattering tensor tomography. 92

ZH zonal harmonics. 62, 63, 65, 66, 82

It is said to be the way of the world that what has long been apart comes together, and what has long been together comes apart.

—*Romance of the Three Kingdoms*

Contents

1

Introduction

Small-angle x-ray scattering (SAXS) is a method for investigating the nanostructure of materials, which probes nanometer-scale variations in the electron density, averaged over the cross-sectional area of the impinging x-ray beam. Advances in synchrotron facilities in recent decades have made SAXS a powerful technique with a wide range of applications. Depending on the size of the x-ray beam, the area of averaging may lie in a range from a few hundred nanometers up to $100 \times 100\,\mu m^2$. SAXS gives statistical information about the size and shape of the scattering objects within the nanostructure of the sample. It is a very suitable method to retrieve the main orientation as well as the orientation distribution from underlying anisotropic ultrastructures, which can be obtained from the anisotropic scattering pattern.

In scanning SAXS (sSAXS), a raster scan of a thin section of a sample is carried out, with a step size which is typically comparable to the beam size. This allows the spatially resolved imaging of the nanostructure of macroscopic samples which may be several millimeters in size, or even larger [1–3]. This is an advantage in particular for hierarchical materials, where structural differences across multiple length scales are of interest [4–6]. This can be compared to direct imaging methods such as scanning electron microscopy (SEM), where probing nanostructural variations in the range 5–300 nm typically limits the field-of-view to approximately 10–100 µm [7]. However, unlike direct imaging methods, sSAXS does not allow for direct retrieval of the real-space electron density. Instead, what is retrieved is the Fourier transform of the autocorrelation function of the electron density in the plane orthogonal to the direction of the impinging beam, known as the reciprocal space map (RSM).

A method that allows for the real-space electron density to be retrieved is x-ray ptychography, where coherent Fraunhofer diffraction is used, and where overlapping domains are scanned [8]. This allows for phase retrieval and direct imaging at very high resolutions, which can be combined with tomography to yield volume-resolved

images of three-dimensional samples [9]. However, like SEM, x-ray ptychography is also limited in its field-of-view when nanometer-scale resolution is desired, to approximately 10–100 µm. Another method for indirectly measuring anisotropic structures allowing for a larger field-of-view is directional dark-field x-ray imaging, which combines x-ray imaging with interferometry. This permits the imaging of anisotropic structures of macroscopic samples, but at relatively large length scales on the order of 1 µm [10].

Thus, sSAXS excels when one desires to probe variations in the average nanoscale structure of a sample over macroscopic regions. In this way, sSAXS is an excellent bridge between larger-scale imaging (*e.g.*, x-ray absorption imaging or visible light microscopy) and very small-scale imaging (*e.g.*, SEM or x-ray ptychography). SAXS can be used to identify regions of interest, which can then be investigated by high-resolution methods. This type of *multi-modular* imaging is an efficient approach to leveraging the complementary advantages of multiple imaging methods while eliminating many of the disadvantages associated with any one method from the final analysis.

Tomography, literally "slice imaging", is a class of methods in which penetrating waves (such as x-rays) are used to probe a sample from multiple angles, and these images are used to reconstruct a volume-resolved image of the sample [11]. SSAXS, when applied to three-dimensional samples rather than thin slices, can be combined with tomographic reconstruction methods to yield a volume-resolved image containing nanostructural information. In cases where the nanostructure is *isotropic*, *i.e.*, when it does not have a preferred orientation, SAXS tomography can be performed using standard methods for *computed tomography* [12–14]. However, for *anisotropic* samples, such as fibrous structures, it is necessary to account for the fact that SAXS only probes the variation in nanostructure in certain directions, orthogonal to the direction of the impinging beam. Standard tomography can be performed of select components of an anisotropic RSM, in particular the component parallel to the axis of rotation, although this discards much of the data [15–17]. It is also possible to compute invariants of the entire probed RSM and reconstruct these, but this generally necessitates the imposition of additional assumptions about the sample [18]. Similarly, it is possible to arrange samples with certain symmetries in the RSM to be invariant to rotation and then perform component-wise reconstruction using standard tomographic techniques [12, 13, 16, 19].

In order to not limit ourselves to these cases, we instead employ a more general approach, small-angle x-ray scattering tensor tomography (SAXSTT), which entails reconstruction of the entire RSM probed by SAXS, and necessitates the tilting of the sample in addition to rotating it [20–22]. SAXSTT introduces a multitude of challenges:

- It is necessary to select a representation for the RSM, which at a single length scale will be isomorphic to a function on the unit sphere, and the data from the detector must be mapped to this representation. Possible choices include polynomial representations (*i.e.*, spherical harmonics), local representations (*i.e.*, radial basis functions defined on a spherical grid), and binning measurements into

regions on the sphere, all of which have advantages and disadvantages.

- Standard scalar tomography algorithms cannot be directly applied to the reconstruction of tensor fields, and designing and implementing robust tensor tomographic algorithms is challenging.
- Tilted measurements introduce additional computational complexity in the real-space part of the reconstruction algorithm, and limit the choice of available implementations for the efficient computation of the real-space line integrals necessary to perform tomographic reconstructions.
- It is not feasible in most cases to measure a sample from all possible angles, which means that there will be artefacts in some areas of the reconstruction (leading to the so-called missing wedge problem), and the analysis should ideally account for this.
- Analysis of the reconstruction, including orientation analysis, peak fitting, background subtraction, and the extraction of various quantities derived from the shape of the SAXS curve, is complicated by a spherical geometry.
- The resultant reconstruction is potentially a highly complex tensor field, which may be difficult to visualize or otherwise get an overview of.

These, and other related complications, make SAXSTT a highly challenging problem at multiple points, from data acquisition to the final analysis. Moreover, we would like to make SAXSTT available to a wider synchrotron user community, which means that these problems must not only be accounted for in a software implementation, but this implementation must be accessible to and usable by a user who is not necessarily an expert in tomography or optimization problems. The relatively large number of parameters needed to specify the experimental setup must be possible to specify and configure in a reasonably intuitive manner, and the implementation should be possible to run without specialized hardware such as high-end graphics cards. Additionally, the practical use of SAXSTT necessitates that the robustness of the technique is thoroughly investigated, so that both users and beamline scientists can have a good idea about the type of information that one can and cannot reliably extract. This includes establishing reference points for the amount and quality of data necessary to reliably carry out reconstructions, in order to minimize acquisition time and maximize the use of allocated beam time. Finally, users must have some access to or be made aware of visualization techniques in order to inspect the data, although the extent to which this is possible to do within the scope of a software project is limited due to the complex and highly specialized nature of three-dimensional rendering techniques.

These considerations provide the motivation for the work presented in this thesis. Chapter 2 outlines the theoretical framework of SAXSTT, including a review of previous

work on SAXS, other approaches to SAXS tomography, and an outline of the problem of representing the RSM. This summarizes the theoretical portions of papers I, II, and IV. Chapter 5 deals with the validation of SAXSTT by demonstrating its robustness, application to simulations, and consistency with tomographic theory. This is treated in papers I and II. Chapter 3 gives an in-depth look at the various reconstruction algorithms utilized and developed over the course of this work, as well as a detailed treatment of the numeric computations necessary to implement them. This is the subject of papers I, II, and IV. Chapter 4 details the Python software MUMOTT, which was developed over the course of this work, and which seeks to address the problem of making SAXSTT available to users. This is the subject of paper IV. Chapter 5 deals with the validation of SAXSTT by demonstrating its robustness, application to simulations, and consistency with tomographic theory. The visualization techniques employed in this work and across all four papers are detailed and discussed in Chapter 6, including various technical considerations due to the large size and atypical structure of tensor tomographic data. Chapter 7 treats the SAXSTT part of paper III, which applies SAXSTT to the study of the chameleon tongue. This chapter details challenges and necessary method improvements in reconstructing this data set, applying q-resolved analysis, segmenting the reconstruction, and performing multi-orientation analysis. The resulting analysis showcases the unique strength of SAXSTT, following the model improvements presented in this work, in resolving complex interwoven and multi-layered ultrastructures. Finally, Chapter 8 concludes the comprehensive summary of this thesis and provides an outlook on opportunities for improvement and future research in the field of SAXSTT.

2

Theoretical framework of SAXSTT

2.1 Small-angle x-ray scattering

SAXS is an experimental method for probing statistical information about the nanostructure of a sample in terms of the RSM, which is the Fourier transform of the autocorrelation function of the electron density in one region of the sample. The way in which SAXS probes the RSM, and how it compares to wide-angle x-ray scattering (WAXS), is best illustrated through the concept of the Ewald sphere shown in Fig. 2.1. In this work we are primarily concerned with scattering at small angles as in the small dark red circle at Fig. 2.1v. The RSM may be used to infer statistical information about the nanostructure of a sample, such as the size and shape distributions of the particles that it is made up of [23]. For samples with anisotropic nanostructures, that is, preferred orientations (see Fig. 2.2), the RSM may be used to deduce information about the orientation distribution function of the anisotropic constituents, such as fibers. We may express the RSM at a given location in a sample as

$$\mathrm{RSM}(\boldsymbol{r}, \boldsymbol{q}) = \iiint \mathrm{d}\boldsymbol{r}' [\tilde{\rho}_N(\boldsymbol{r}' - \boldsymbol{r}) \exp(-i\boldsymbol{q} \cdot (\boldsymbol{r}' - \boldsymbol{r}))], \tag{2.1}$$

where N denotes a neighbourhood around $\boldsymbol{r}$, $\tilde{\rho}_N(\boldsymbol{r}' - \boldsymbol{r})$ is the auto-correlation function of the electron density over N at $\boldsymbol{r}' - \boldsymbol{r}$, and $\boldsymbol{r}'$ is the integration variable. Specifically, we can write

$$\tilde{\rho}_N(\boldsymbol{r}) = \iiint \mathrm{d}\boldsymbol{r}' \left[w_N(\boldsymbol{r}') \rho(\boldsymbol{r}' - \boldsymbol{r}) \rho(\boldsymbol{r}') \right], \tag{2.2}$$

where $w_N(\boldsymbol{r}')$ is a windowing function centered on the neighbourhood N, encoding the point spread function of the beam and having a small thickness in the direction of beam propagation.

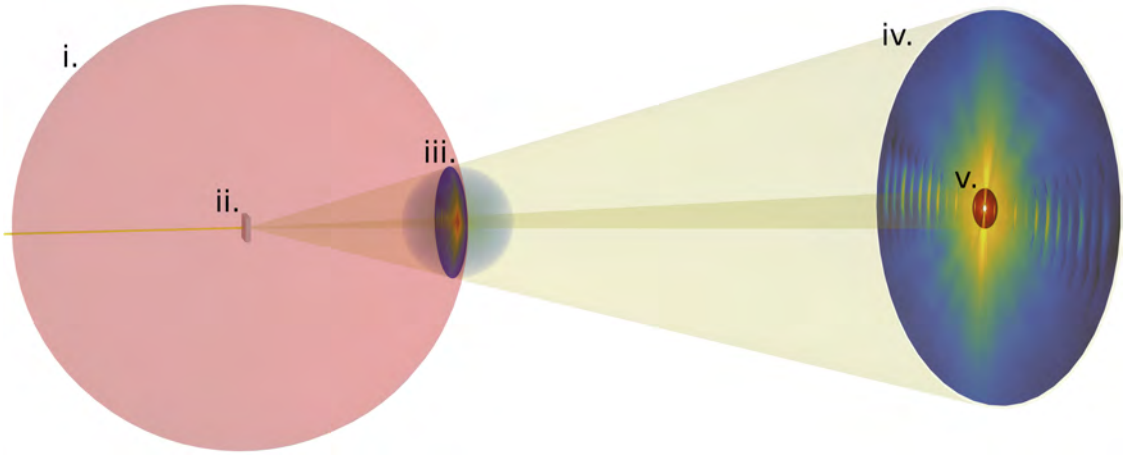

Figure 2.1: Simulation of the effect of the curvature of the Ewald sphere (large red sphere at **i.**), for when a beam impinges on a sample (small object at **ii.**). The edge of the Ewald sphere intersects with the origin of the RSM (transparent small sphere at **iii.**) and the cut of the intersection forms the probed part of the RSM, a curved disk. The outer part of the cut scatters at wide angles, and is greatly magnified when projected onto the detector (large blue-green-yellow **iv.**). A single circle on the detector forms a small circle of one shell of reciprocal space due to the curvature of the Ewald sphere. The inner part of the cut (dark red area at **v.**) is much less dramatically magnified as the scattered rays are nearly parallel due to the small scattering angle. A single circle in this part of the detector approximately forms a great circle of a small shell in reciprocal space, due to the small curvature of the Ewald sphere over the angle.

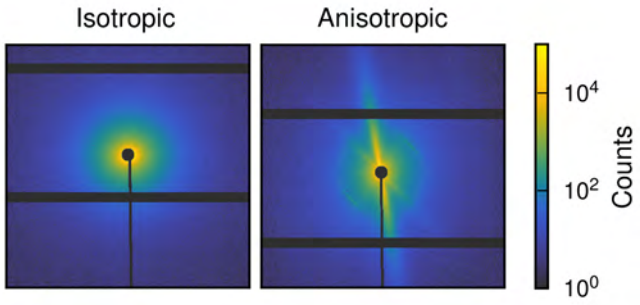

Figure 2.2: Examples of an anisotropic and an isotropic RSM, measured by scanning SAXS.

This description is complicated by the fact that theoretical models used to describe the RSM assume that the real-space structure is an infinitely extended, repeating unit cell. The justification for this deviation from the theory is that the highly monochromatic beams with a long correlation length that can be obtained at synchrotrons are in effect plane waves at typical SAXS nanostructure length scales of 1 – 300 nm. The beams are thus local on the scale of the sample as a whole, but have a very large extent on the

scale of the nanostructural elements.

To begin discussion our measurement of the RSM, we note that a beam impinging upon, passing through, and exiting a non-emissive slab of material obeys the basic equation

$$I_0 = I_T + I_S + I_A,$$

where I_0 is the impinging intensity while I_T, I_S, and I_A are transmitted, scattered, and absorbed intensity, respectively. For a thin slab we may then write, using the fact that under these constraints the total intensity I_0 is constant as the beam passes through the material, or in other words, $\mathrm{d}I_0 = 0$,

$$-\mathrm{d}I_A = \mathrm{d}I_T + \mathrm{d}I_S.$$

Assuming that the beam does not couple strongly to the material, *i.e.*, that it has a sufficiently high energy, we may neglect dI_S in relation to dI_T and write

$$-\mathrm{d}I_A \approx \mathrm{d}I_T.$$

Assuming linear absorption such that $I_T(s + ds) = I_T(s)(1 - a\,ds)$, we integrate from 0 to s, where s is the thickness of the slab, to obtain that

$$I_0 - I_A \approx I_T = I_0 \exp(-as). \tag{2.3}$$

Moving on to the previously neglected scattering term, we note that the scattered intensity increases as parts of the transmitted intensity are scattered, and decreases as parts of the scattered intensity are absorbed. Assuming the scattering angle to be small ($\cos(\theta) \approx 1$), the absorption coefficient will be the same a as for $I_T(s)$, and the path length will also be the same. In other words, we can approximate

$$\begin{aligned} dI_S &\approx ds(\beta I_T(s) - aI_S(s)) \\ &\approx ds(\beta I_0 \exp(-as) - aI_S(s)). \end{aligned}$$

Because we assume the scattering coefficient β to be small ($s\beta \ll 1$), we neglect higher-order scattering (*i.e.*, terms proportional to $\beta I_S(s)$, $(\beta I_S(s))^2$, and so on). Knowing the scattered intensity at $s = 0$ (the point of impingement) is $I_S(0) = 0$, this differential equation has the solution

$$I_S(s) = s\beta I_0 \exp(-as).$$

Now, we find that

$$\frac{I_S(s)}{I_T(s)} = s\beta.$$

We have carried out this entire derivation on a thin slab of material (assumed to have a constant scattering coefficient), and we have considered the total amount of scattering across all angles. We will now extend this to scattering through a thicker slab, which may not have a constant scattering potential, and parameterize the scattering in terms of the *reciprocal space vector* $\boldsymbol{q}$. Assuming the wavelength of the beam is much smaller than the characteristic length scales of the electron density of the material, while the beam is effectively a plane wave at those length scales, we may now express the scattering-transmission quotient in terms of the RSM, that is,

$$\frac{I_S}{I_T}(s + \mathrm{d}s, \boldsymbol{q}) - \frac{I_S}{I_T}(s) = \textsc{rsm}(\boldsymbol{r}_0 + s\hat{\boldsymbol{s}}, \boldsymbol{q})\mathrm{d}s,$$

where $\boldsymbol{r}_0$ is the point of impingement. The size of the neighbourhood N in Eq. (2.1) is determined by the size of this impinging beam.

This differential equation is solved by integrating both sides, yielding

$$\frac{I_S}{I_T}(s_{\text{end}}, \boldsymbol{q}) = \int_{s_0}^{s_{\text{end}}} \textsc{rsm}(\boldsymbol{r}_0 + s\hat{\boldsymbol{s}}, \boldsymbol{q})\mathrm{d}s. \tag{2.4}$$

In other words, the transmission-normalized scattering measured is the line integral of the RSM taken over the path of the beam. The measurable portion of the RSM is a two-dimensional slice through the plane orthogonal to the impinging beam.

2.2 Tomography of anisotropic SAXS signals

Tomography includes a large number of imaging techniques which have in common that they enable a sample to be studied in a slice-by-slice manner by probing it using a penetrating wave such as radio waves, visible light, x-rays, neutrons or sound. We are specifically interested in tomographic reconstruction of projection measurements that can be modelled as line integrals of three-dimensional fields. Such projections are described by the John transform, also known as the x-ray transform [11], which has the general form

$$P[f](L) = \int_{-\infty}^{\infty} \mathrm{d}t\, f(\boldsymbol{x}_0 - t(\boldsymbol{x}_0 - \boldsymbol{x}_1)), \tag{2.5}$$

where f is a field in three-dimensional space, L is a line in three-dimensional space, and $\mathbf{x_0}$ and $\mathbf{x_1}$ are two distinct points on L. A transformed field obeys John's equation, an ultra-hyperbolic differential equation, which may be used to derive consistency conditions for projection data, which determine the extent to which the original field can be reconstructed from the data. Since the RSM in Eq. (2.4) is taken along a line integral, it is clear that under at least some conditions it can be subjected to tomographic

reconstruction. At any constant $\|\mathbf{q}\|$ where the RSM does not depend on the direction of $\boldsymbol{q}$, it is effectively a scalar field, and it can therefore be reconstructed using standard tomographic methods [16, 17, 24]. This becomes more complicated if the RSM is anisotropic, *i.e.*, depends on the direction of $\mathbf{q}$. If an axis exists such that the cut of the RSM of each volume element is invariant with respect to rotation about it, then a tomographic dataset can be collected by rotating the sample about this axis and treating the RSM at each angle as a separate scalar field [12, 13, 18, 19].

The issue is substantially complicated when no simple condition of invariance exists for the RSM. It is helpful to begin to consider what happens if we attempt the collection of an ordinary tomographic dataset by rotating about some arbitrary axis $\hat{\boldsymbol{\imath}}$. We obtain

$$D(j,k,\alpha,q,\varphi) = \int_{-\infty}^{\infty} \mathrm{d}t \, \mathrm{RSM}(\boldsymbol{\nu}(j,k,\alpha) + t\hat{\boldsymbol{p}}(\alpha), \|q\| \, (\hat{\boldsymbol{\imath}} \cos\varphi + \hat{\boldsymbol{\imath}} \times \hat{\boldsymbol{p}}(\alpha) \sin\varphi)),$$

where (j,k) are two coordinates in the plane of projection, α is the angle of rotation about $\hat{\boldsymbol{\imath}}$, φ is an azimuthal angle on the detector, $\boldsymbol{\nu}$ is a vector which locates some point in the plane of projection, and $\hat{\boldsymbol{p}}(\alpha)$ is the direction of projection. Two crucial conclusions can be drawn from this equation. First, for $|\cos\varphi| = 1$, the reciprocal space vector does not depend on α. Second, for $|\cos\varphi| \neq 1$, it instead traces out a cone in reciprocal space (a circle on the sphere of constant $\|\boldsymbol{q}\|$). This means that for a given detector direction, we measure a different direction in reciprocal space in every unique projection direction except for the detector direction parallel to the axis of rotation.

We can understand this in the following way: for the components of the RSM parallel to the axis of rotation, we have an ordinary tomographic dataset. For all other components, we only know the *average value* of the RSM along that particular direction. It would be possible to rotate the sample around multiple axes in sequence, reconstruct the parallel component along each of these axes, and then interpolate the rest of the RSM. However, this would be somewhat wasteful, because we would be throwing away most of the data, except along those individual axes. One can attempt a measurement array where multiple subsets of the measurements line up with a great circle; this is the so-called *virtual axis* approach of Schaff *et al.* (2015) [20]. However, if we assume that there is some amount of continuity in the RSM, we no longer have to think in terms of individual axes of invariance. Instead, for a given direction in reciprocal space, we can compute the tomographic trajectory along which we would have to measure in order for that direction to be an invariant, and then see if we have measurements in the neighborhood of that trajectory. That is to say, rather than measuring with multiple main rotation axes (or in a way that takes virtual axes into account), we try to measure as densely as needed across as much of the hemisphere as possible. A relatively simple way of doing this is to use a tilt axis, orthogonal to the main rotation axis, see Fig. 2.3. It is possible to cover the hemisphere up to a latitude of approximately 45° in this way, before the rotation stage begins to obstruct the x-ray beam. This rotation-and-tilt approach was taken by Liebi *et al.* (2015, 2018), in addition to Schaff *et al.* (2015)

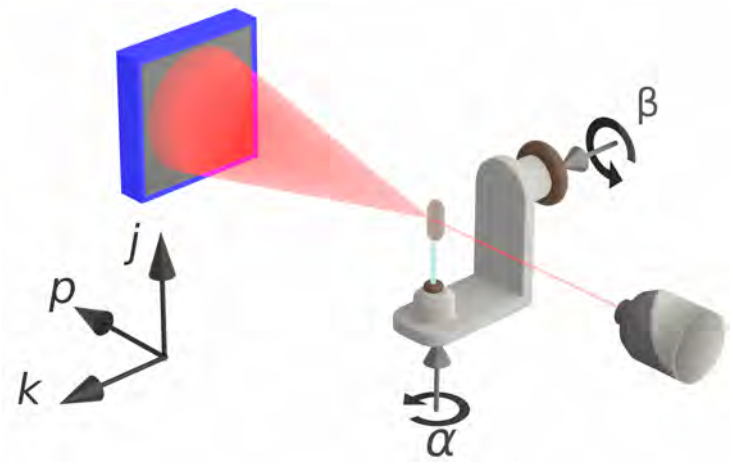

Figure 2.3: Schematic of experimental setup for measurements with a rotation and tilt axis. The beam travels in the $\hat{\boldsymbol{p}}$ direction, and the sample is moved in the $\hat{\boldsymbol{j}}$ and $\hat{\boldsymbol{k}}$ directions to carry out scanning SAXS. The sample can be rotated about the $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}$ axes to scan it from different angles.

[20–22]. Assuming that measurements are made in a sufficiently dense fashion such that the measurement directions cover the hemisphere up to 45°, this should allow for reasonably good reconstruction of reciprocal space components within a 45° cap. For other directions, the data set will only be partial — there will be *missing wedges* in these directions. We can express the two-axis transform in the following way,

$$\boldsymbol{D}(j,k,\alpha,\beta,q,\varphi) = \int_{-\infty}^{\infty} \mathrm{d}t\, \mathrm{RSM}\left(\boldsymbol{v}(j,k,\alpha,\beta) + t\hat{\boldsymbol{p}}(\alpha,\beta), \boldsymbol{q}(\alpha,\beta,\varphi)\right),$$

where $\boldsymbol{q} = q\hat{\boldsymbol{q}}(\alpha,\beta,\phi)$, where $\hat{\boldsymbol{q}}$ gives the direction in q-space as a function of rotation, tilt, and detector angles. However, it is not practical to directly write out the expressions for the effects of modifying (α,β) (but see Sect. 4.1.1 for a discussion of how this and other geometry-related concerns are handled in Mumott). Thus, for a more compact representation, we will instead express the orientation of the sample using a single variable s, which we take to parameterize a trajectory on the sphere of rotation that depends on (α,β),

$$\boldsymbol{D}(j,k,s,q,\varphi) = \int_{-\infty}^{\infty} \mathrm{d}t\, \mathrm{RSM}\left(\boldsymbol{v}(j,k,s) + t\hat{\boldsymbol{p}}(s), \boldsymbol{q}(s,\varphi)\right). \tag{2.6}$$

In Fig. 2.4 we see two components of the tensor John transform of a field of spherical functions with respect to rotation about two orthogonal axes. The amplitudes in the directions which are aligned with the axes change more smoothly as the projection number changes. Moreover, in the orthogonal direction, some lines vanish as the rotation angle changes, which is easiest to see on the left-hand side of panel b). The sum

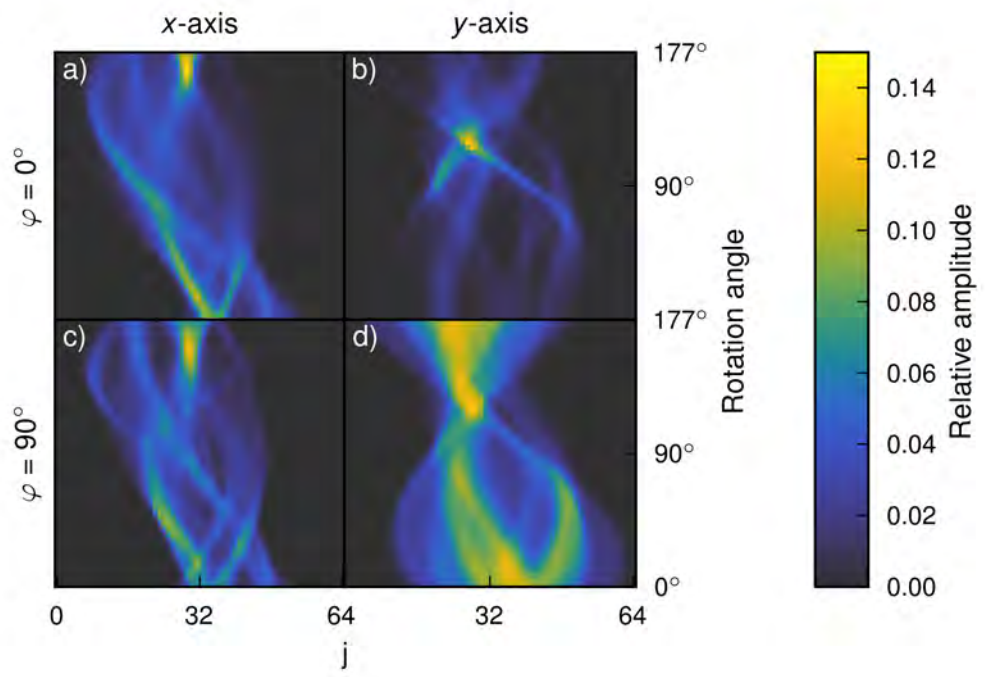

Figure 2.4: Tensor John transform of a field with respect to rotation about two orthogonal axes and two measured directions on the detector. The amplitudes have been normalized by the sum of each horizontal line. The vertical axes show the rotation angle while the horizontal axis shows a single scanned point across one line. The $\varphi = 0^\circ$ direction in panels **a)** and **b)** is aligned with the x-axis and the 90° direction in panels **c)** and **d)** is aligned with the y-axis. The amplitude of the transform with respect to rotation about the x-axis in panels **a)** and **c)** changes more smoothly in the $\varphi = 0^\circ$ direction when the rotation angle changes than in the $\varphi = 90^\circ$ direction. Conversely, the transform with respect to rotation about the y-axis in panels **d)** and **b)** is more continuous in the $\varphi = 90^\circ$ direction.

of the amplitude over each horizontal line is constant when the detector direction is parallel to the rotation axis, because the same direction is always being measured in three-dimensional reciprocal space, and the same part of the sample is always in view. This is not true when a non-parallel component is being measured, since the direction in three-dimensional reciprocal space will then change. Thus, normalizing by this sum emphasizes the lack of smoothness with respect to change in the rotational angle.

There are limitations to the rotation-and-tilt approach, since it depends on the possibility of interpolating the RSM between measured directions. If, for example, the RSM contains very sharp features along certain directions, *e.g.*, due to very precise alignment of fibrils, some of those peaks might only be visible at one particular sample orientation. In this case, it may be necessary to make additional measurements at neighbouring orientations in order to determine the amplitude, width, and position of these peaks

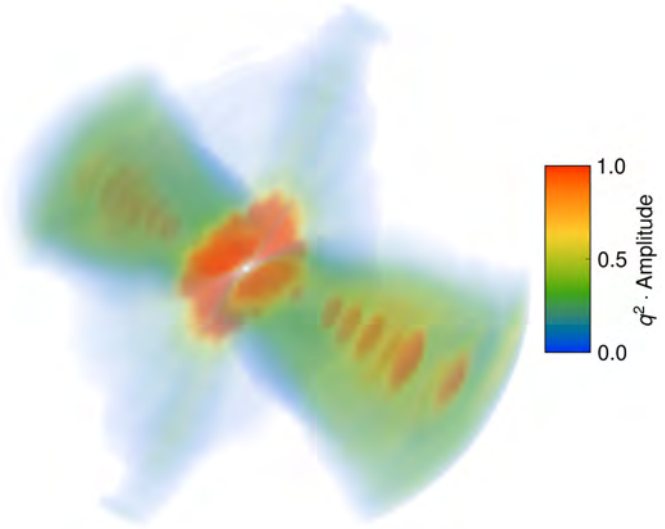

Figure 2.5: Kratky render of three-dimensional anisotropic RSM from a collagen-rich region in a chameleon tongue. The q-range shown in this reconstructed RSM is approximately 0.02 – 1.6 nm^{-1}, corresponding to a d-spacing of about 3.75 – 300 nm. The large disk in the center corresponds to equatorial scattering from the diameter of collagen fibrils, whereas the multiple smaller disks correspond to meridional scattering from the longitudinal spacing of collagen molecules.

accurately. In order to better understand these issues it is necessary to discuss how we represent the RSM.

2.3 Representation of the reciprocal space map

The RSM is a function of the three-dimensional reciprocal space vector $\boldsymbol{q}$, which is measured in two dimensions. See Fig. 2.5 for an example of a three-dimensional RSM. In this case, the three-dimensional map has been interpolated from 194 spherical shells. A set of 10 such shells is shown in Fig. 2.6. Because a given magnitude of the reciprocal space vector corresponds to a given length scale in real space, per the Laue equation or the equivalent Bragg condition [25], it is natural to parameterize reciprocal space in spherical coordinates. Consequently, when discretizing the RSM it is reasonable to begin with dividing the three-dimensional RSM into spherical shells. This means that the problem of representation mainly concerns representing functions on the unit sphere. Similarly, this means that the data reduction becomes an issue of mapping data measured on a two-dimensional grid of pixels onto a representation in polar coordinates.

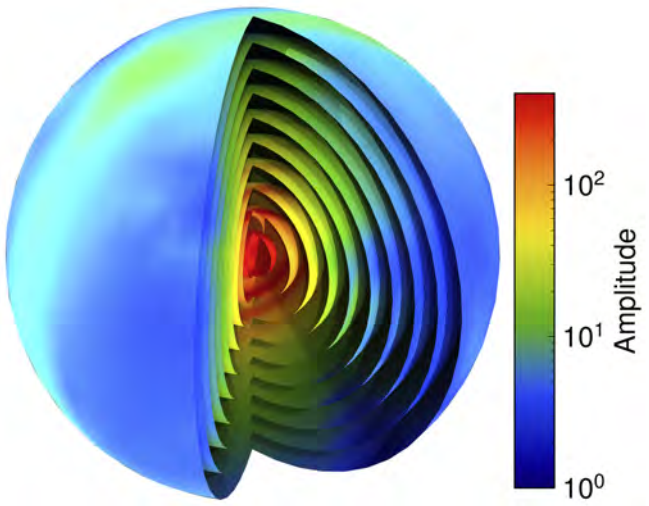

Figure 2.6: 10 RSM shells of a collagen-rich region in a chameleon tongue. Each of these shells are reconstructed separately.

2.3.1 Data reduction

A SAXSTT data set is very large. Consider a measurement after a fairly modest scheme, consisting of 50×50 scan points per frame, with 200 frames measured at different sample orientations, and with 4 megapixels per diffraction image measured. If each measured pixel is represented by a 16-bit, or 2-byte value, we end up with $4 \times 10^6 \times 50 \times 50 \times 200 \times 2\,\mathrm{B} = 4\,\mathrm{TB}$ of data. If we increase each scanning dimension, and the number of frames, by a factor of 2 the data size increases to 32 TB. These data sizes are not manageable by ordinary means, and the data must be drastically reduced. The obvious targets for reduction are the diffraction patterns themselves.

At small scattering angles, diffraction patterns generally follow power laws with respect to $\|\boldsymbol{q}\|$. This means that the scattering intensity contributions from various structures follow $I \approx a\|\boldsymbol{q}\|^b$ with some coefficient a and exponents b. Hence, it is natural to study $\log I \approx \log a + b \log(\|\boldsymbol{q}\|)$. Different contributions dominate at different scales. We thus generally want to reduce data in such a way that we have a high resolution for small $\|\boldsymbol{q}\|$ and conversely a low resolution when $\|\boldsymbol{q}\|$ is large. This is in addition motivated by the desire to reduce noise levels by including more data points at higher $\|\boldsymbol{q}\|$, where the intensity is typically several orders of magnitude lower than at lower $\|\boldsymbol{q}\|$.

The most common approach to reducing diffraction patterns is radial-azimuthal integration (sometimes just called radial integration or azimuthal integration). In this approach, the pixels are first given coordinates in terms of $(\|\boldsymbol{q}\|, \varphi)$. A set of radial bins $[\|\boldsymbol{q}\|_h, \|\boldsymbol{q}\|_{h+1})$ and azimuthal bins $[\varphi_i, \varphi_{i+1})$ are then defined, and the pixels are sorted into the two-dimensional array $\boldsymbol{I}_{hi}$ implicitly defined by these binnings. The

array is normalized by the number of pixels in each bin, N_{hi}[1]. The principal advantage of this method is that the uncertainty of each measurement can easily be tracked. If the bin is sufficiently small, the uncertainty is, for single-photon counting detectors, $\sqrt{I_{hi}}/N_{hi}$. For a larger bin with many pixels that covers a large angle or a large q-range, the uncertainty can be estimated as the standard deviation of the intensities of all pixels sorted into the bin. It is also possible to use whichever of these uncertainties is larger as the estimate.

There are two other options in terms of reducing the data. One option is to define a grid of nodes, rather than of bins, and to use a resampling method of choice (*e.g.*, Lanczos resampling) — effectively, one needs to run a convolution kernel over each image and then sample at pre-defined points. Another option is to fit the entire diffraction pattern to a set of basis functions with the appropriate symmetries such as the Zernike polynomials. In this case, the method of fitting or resampling would need to be done in a way that minimizes the uncertainty due to noise. In addition, constraints such as non-negativity would need to be imposed. Because of the relative complexity of such alternative schemes, in this work, I have used the traditional approach of radial-azimuthal integration.

2.3.2 Functions on the sphere

There are two aspects to functions on the sphere in the context of SAXSTT. One aspect is the general question of how to expand and analyze a function on the sphere in a discrete, finite representation. The other aspect is how to map detector measurements to the spherical representation, *i.e.*, how to interpolate spherical functions from measurements on great circles.

2.3.2.1 Basis sets

To first cover the issue of how a function on the unit sphere, corresponding to one $\|\boldsymbol{q}\|$-bin on the detector, is best represented, we consider that such a function is canonically parameterized by a polar and an azimuthal angle, which we will denote (θ, ϕ), respectively. We are not exclusively interested in strict basis sets for functions on the unit sphere, but also in approximate representations, *i.e.*, vector-valued functions of coordinates on the sphere which satisfy

$$f(\theta, \phi) \approx \sum_i \boldsymbol{a} \cdot \boldsymbol{B}(\theta, \phi),$$

where $f(\theta, \phi)$ is any sufficiently smooth function on the unit sphere, a_i is element i of some vector of coefficients, and $\boldsymbol{B}_i$ is element i of a vector-valued function of (θ, ϕ). We

[1] This enumeration includes only *valid* pixels.

do not necessarily want this approximate equality to become exact. Rather, we would in many cases prefer our representation $\boldsymbol{a}$ to yield a less noisy and smoother function. In fact, it may be more natural to understand our approach as solving for a *projection* of $f(\theta, \phi)$ onto a representation-dependent subspace of the unit sphere, where the type of representation determines the properties of this subspace.

If we forget momentarily about the tomography aspect of the problem that we are studying, the projection of a function on the sphere into a subspace spanned by a basis set may be written as

$$f_B(\theta, \phi) = \boldsymbol{B}(\theta, \phi) \cdot \underset{\boldsymbol{a}}{\operatorname{argmin}} \left[\int \mathrm{d}\Omega \, |f(\theta, \phi) - \boldsymbol{a} \cdot \boldsymbol{B}(\theta, \phi)|^2 \right] \tag{2.7}$$

The subscript $\boldsymbol{B}$ in $f_B(\theta, \phi)$ indicates that this is the projection of the function into the subspace spanned by $\boldsymbol{B}(\theta, \phi)$. In fact, this type of function representation is already a type of reconstruction and should be accompanied by regularizing constraints on, *e.g.*, the sparseness, smoothness, or sign of the vector $\boldsymbol{a}$. The constraint would depend on the function in question, but we can write it as

$$f_B^{\Lambda}(\theta, \phi) = \boldsymbol{B}(\theta, \phi) \cdot \underset{\boldsymbol{a}}{\operatorname{argmin}} \left[\int \mathrm{d}\Omega \, |f(\theta, \phi) - \boldsymbol{a} \cdot \boldsymbol{B}(\theta, \phi)|^2 + \Lambda(\boldsymbol{a}) \right], \tag{2.8}$$

where Λ is a scalar-valued function of the vector spaced spanned by $\boldsymbol{B}$ such as a norm or an indicator function. We can then understand f_B^{Λ} as the most proximate function to the projection of f onto the subspace spanned by $\boldsymbol{B}$ which satisfies the constraint implied by Λ. We will return to this expression after considering the details of the representations.

It is natural to consider a polynomial representation — *i.e.*, the *spherical harmonics*. There are clear advantages to this representation. It consists of orthogonal basis functions, it is invariant with respect to rotation, there are many theorems that can be used to compute statistics and properties of the represented function (such as the mean amplitude and principal orientation), and it is easy to apply filters using the spherical harmonic convolution theorem, to name a few. On the other hand, spherical harmonics also have drawbacks. They suffer from the Gibbs phenomenon (ringing artefacts), it is difficult to enforce bounds (such as positive definiteness) on the function, and they do not yield sparse representations for data that is sparse on the sphere.

The chief alternative to a spherical harmonic representation is a representation based on a *grid of basis functions*. Wavelets, which are local functions frequently used on the plane, are not as useful on the sphere, as the various theorems which motivate the use of wavelets do not apply on the sphere. A more promising alternative are non-negative radial basis functions, such as Gaussians. Such a basis will yield smooth functions with some denoising properties since neighbouring basis functions have a significant overlap. Moreover, each basis function is local, roughly meaning that a function, which is non-zero over only a small contiguous area, can be represented sparsely, *i.e.*, with few

non-zero coefficients. This is a useful property in many optimization problems. They can be converted to spherical harmonics by evaluating the function on a sufficiently dense grid (given by the Driscoll-Healy sampling theorem [26]) in order analyze derived properties such as mean amplitude, variance or orientations. The Gaussian radial basis functions are discussed at further length in Sect. 3.2.1.

It would be possible to bin measurements using nearest-neighbour interpolation on a grid. Essentially, this is using a step function as the basis function. This representation has advantageous properties such as orthogonality (which Gaussian basis functions lack) and more easily enforced sparsity, but lacks smoothness and requires additional processing, such as a low-pass filter, to be represented as a three-dimensional surface. While they are implemented in MUMOTT they are not considered at length in this work.

Another possibility are barycentric or bilinear interpolations, which yield a representations that are piecewise smooth and everywhere continuous. They thus are interesting alternatives but require additional computations and in the case of bilinear interpolation, they limit the choice of a grid. For these reasons they have not been used in this work.

Finally, it is useful to discuss symmetry-constrained spherical harmonics, such as zonal harmonics. Such a representation is used in Liebi *et al.* (2015, 2018) [21, 22], and has certain interesting properties. In particular, its enforced symmetry can fill in missing data, which can in principle improve a reconstruction if the symmetry applies. However, it is substantially harder to solve optimization problems using such a representation. This is because the zonal harmonics do not form a linear subspace of the spherical harmonics , *i.e.*, two zonal harmonics with different main orientations do not sum to a third spherical harmonic if the band-limit $\ell_{\max} > 2$. This requires optimization for an additional orientation vector or equivalently two orientation angles. Because only subspaces of the spherical harmonic with adjacent orientations can be accessed in each iteration, the optimization is susceptible to getting trapped in local minima. There are potentially interesting applications for enforced symmetries of this type, such as using them in conjunction with other representations as soft constraints. While this type of representation is also implemented in MUMOTT they are not considered in detail outside the context of the robustness study that is the topic of paper I.

2.3.2.2 Detector-to-sphere mapping

The second issue of representation is the matter of how to map binned functions on the detector to functions on the sphere. Effectively, the binning of measurements into azimuthal bins on the detector corresponds to a line integral carried out along an arc on the sphere, divided by the length of the arc. In fact, these line integrals can also be treated as the projection of a function into a subspace spanned by basis functions which are zero everywhere except along these arcs, assuming that the arcs do not overlap. This is because the minimizer of the integrated squared difference between a function and a constant is the mean value of that function, satisfying the optimization problem in the

projection operation given in Eq. (2.7). This means that we can write the problem of mapping functions between the detector segments and the sphere as finding a vector $\boldsymbol{a}^*$ that satisfies

$$\boldsymbol{a}^* = \underset{\boldsymbol{a}}{\operatorname{argmin}} \left[\sum_h \int_{\hat{l} \in D_h(\hat{l}) \neq 0} \mathrm{d}\hat{l} \left| f_{Dh} \boldsymbol{D}_h \left(\hat{l}\right) - \boldsymbol{a} \cdot \boldsymbol{B} \left(\hat{l}\right) \right|^2 + \lambda \left\| a \right\|_1^1 \right]. \tag{2.9}$$

Here, $\boldsymbol{D}_h$ is an indicator function which is 0 outside the arc and $\frac{1}{L_h}$ on the arc, where $\boldsymbol{L}_h$ is the arc length. $\lambda \|a\|_2^2$ is a regularization term with λ being the regularization coefficient[2]. The regularization term ensures a sparse solution to the projection. In other words, the way in which we attempt to reconstruct a function that was projected onto $\boldsymbol{D}$, the detector basis, in $\boldsymbol{B}$, the reconstruction basis, is to find a sparse function in the reconstruction basis which yields the same projection onto $\boldsymbol{D}$ as the original function.

No matter how we frame the issue, in the end the mapping will necessitate evaluating a term of the form

$$\boldsymbol{M}_{ih} = \int_{\hat{l} \in D_h \neq 0} \mathrm{d}\hat{l} \boldsymbol{D}_h \left(\hat{l}\right) \boldsymbol{B}_i \left(\hat{l}\right), \tag{2.10}$$

which appears if we take the gradient of Eq. (2.9) with respect to either f_{Dh} or a_i. We address the evaluation of this term, which can be understood as the overlap between the two basis functions, in greater detail in Chapter 3.

In a SAXSTT problem, we will look for an $\boldsymbol{a}$ that simultaneously satisfies many constraints like Eq. (2.9). However, if our set of measurements is sparse enough and our basis $\boldsymbol{B}$ has a sufficiently high resolution, this may not be enough. In this case, we may wish to introduce additional constraints, which this framing gives a natural way to do. One approach would be to introduce a different kernel. This can be motivated as follows: assume that the original function f is smooth enough that

$$\int \mathrm{d}\Omega f \left(\hat{l}\right) \boldsymbol{D}_i \left(\hat{l}\right) \approx \int \mathrm{d}\Omega f \left(\hat{l}\right) \boldsymbol{E}_i \left(\hat{l}\right) \tag{2.11}$$

In this case, if the line segment kernel $\boldsymbol{D}_i$ does not result in enough coverage for reconstruction, we can use the alternative kernel $\boldsymbol{E}_i$ (which might be a surface kernel with a width, *e.g.*, a line kernel convolved with a Gaussian). Alternatively, one could in principle write the problem along the lines of Eq. (2.8). That is, rather than using the matrix in Eq. (2.10) to propagate changes to the sphere, one could evaluate Eq. (2.9) with further constraints, such as smoothness, applied to the function represented by

[2]In practice, λ is typically very small and the regularization term is not explicitly considered. It simply encodes an expectation that the mapping is as sparse as possible in **B**-space. When consider this problem more generally, this leads to the topic of *basis pursuit*.

$\boldsymbol{a}^*$. This would, however, significantly complicate the problem, as it would no longer be possible to precompute a matrix according to Eq. (2.10) and carry out the computations using this matrix. The details and potential use of this possibility will be discussed in greater detail in Chapter 3.

3

SAXSTT Algorithms

3.1 SAXSTT as an optimization problem

In SAXSTT, we wish to reconstruct a discretized approximation of a six-dimensional field $f(\boldsymbol{r}, \boldsymbol{q})$. In order to discretize the $\boldsymbol{q}$-dependent part of the function, we divide the six-dimensional function into a set of three-dimensional scalar fields $a_i(\boldsymbol{r})$ and basis functions $\boldsymbol{B}_i(\boldsymbol{q})$. The field has been subjected not only to the John transform Eq. (2.6), but also to integration over the sphere (Sect. 2.3.2), which we parametrize with a mapping $\boldsymbol{D}_h(s, \boldsymbol{l})$. Thus we are, roughly speaking, concerned with solving the inverse problem

$$\textsc{rsm}_h(s, j, k) \approx \int_{D_h(s,\boldsymbol{l})\neq 0} \mathrm{d}\boldsymbol{l}\, D_h(s, \boldsymbol{l}) B_i(\boldsymbol{l}) \int_{-\infty}^{\infty} \mathrm{d}t\, a_i\left(\boldsymbol{v}(s, j, k) + t\boldsymbol{p}(s)\right) \tag{3.1}$$

using the Einstein summation convention for repeated indices. The left-hand side, $\textsc{rsm}_h(s, j, k)$ side is our measured data, and the right-hand side models

$$\textsc{rsm}_h(s, j, k) = \int_{D'_h(s,\boldsymbol{l})\neq 0} \mathrm{d}\boldsymbol{l}\, D'_h(s, \boldsymbol{l}) \int_{-\infty}^{\infty} \mathrm{d}t\, f(\boldsymbol{v}(s, j, k) + t\boldsymbol{p}(s), \boldsymbol{l}). \tag{3.2}$$

That is to say, we are probing the 6-dimensional field $f(\boldsymbol{r}, \boldsymbol{q})$. The real-space part of the field is integrated along directions $\boldsymbol{p}(s)$, where s parameterizes the set of integration directions. The integration is additionally parameterized by a plane coordinate, $\boldsymbol{v}(s, j, k)$, where j and k are two Cartesian coordinates in the plane orthogonal to the direction of integration. The reciprocal-space part of the field is integrated and averaged over kernels $\boldsymbol{D}_h(s, \boldsymbol{l})$ in q-space, following the data reduction scheme in Sect. 2.3.1. The theoretical kernel $\boldsymbol{D}'_h(s, \boldsymbol{l})$ is not exactly the same as the model kernel — the theoretical kernel includes discretization noise from the detector and averaging over a $\boldsymbol{q}$-range, whereas we only consider spherical shells of $\boldsymbol{q}$ in our model. Moreover, as discussed in Sect. 2.3.2,

we may sometimes wish to use a modified kernel in order to impose assumptions or constraints, although we do not here cover the possibility of using the more general mapping constraints of Eq. (2.8).

It is difficult to gain an overview of this problem due to the many variables and potential configurations. We will start by discussing the characteristics of the field $f(\boldsymbol{r}, \boldsymbol{q})$. This function is essentially the RSM given in Eq. (2.1). However, we must be somewhat careful in this discussion. The definition of the RSM includes a Fourier transform and an auto-correlation function computed over a local neighbourhood, see Eqs. (2.1) and (2.2). It is advantageous to the formulation of the problem of SAXSTT to let this scale be defined by the size of the impinging beam, assuming that the beam is symmetrical enough. In this case, we can speak of measuring the RSM at one particular point rather than measuring the average RSM over an area, because we are talking about a field which is the result of an integral over the probed region. Moreover, the inclusion of this kernel makes the RSM a smooth function, which is advantageous when considering solution algorithms. If the scale of the RSM neighbourhood is defined differently, such as by the smallest size that makes physical sense (which would be twice the length scale and thus $\boldsymbol{q}$-dependent), we must admit another layer of approximation and we run into complications relating to, *e.g.*, the coherence length of the beam, which must be accounted for. This is because our data undergoes convolution with a beam kernel, which makes it more difficult to compute the integral accurately. The only disadvantage of this framing is that two measurements carried out with the same parameters except for the beam size will measure different RSMs, which may seem counter-intuitive. On the other hand, the difference between the two data sets will be the same convolution regardless of the framing. With this in mind, we prefer to think of measurements being carried out *at* each real-space coordinate, rather than being integrated over a neighbourhood.

There are other issues associated with the way our problem is posed. On one hand, it is certainly possible to approximately solve Eq. (3.1) for some field $\boldsymbol{a}(\boldsymbol{r})$, given $\boldsymbol{B}(\boldsymbol{q})$, $\boldsymbol{D}(s, \boldsymbol{q})$ as well as some reasonably efficient way to approximate the John transform. On the other hand, there is no automatic guarantee that the solution $\boldsymbol{a}(\boldsymbol{r}) \cdot \boldsymbol{B}(\boldsymbol{q})$ will be a "good" approximation to $f(\boldsymbol{r}, \boldsymbol{q})$, in the sense that the relationship

$$\boldsymbol{a}(\boldsymbol{r}) = \underset{\boldsymbol{a}(\boldsymbol{r})}{\operatorname{argmin}} \iint \mathrm{d}V\mathrm{d}Q \, |\boldsymbol{a}(\boldsymbol{r}) \cdot \boldsymbol{B}(\boldsymbol{q}) - f(\boldsymbol{r}, \boldsymbol{q})|^2$$

is not necessarily satisfied (approximately or exactly) for the total squared difference or some similar measure. Much of optimization theory is concerned with formulating optimization problems such that conditions of this type are better satisfied, under some assumptions about $f(\boldsymbol{r}, \boldsymbol{q})$ being a sufficiently "nice" function (smooth, noise-free, having compact support, etc). Alternatively, one can regard it as solving for proximity

to a function f^* that satisfies

$$f^* = \underset{g}{\operatorname{argmin}} \iint \mathrm{d}V\mathrm{d}Q \, |g(\boldsymbol{r}, \boldsymbol{q}) - f(\boldsymbol{r}, \boldsymbol{q})|^2 + \Lambda(g(\boldsymbol{r}, \boldsymbol{q})),$$

where Λ is a constraint function and f^* is a type of generalized projection of f. In this case, the assumptions encoded in Λ do not necessarily have to hold for f in order for the optimization problem to be coherently posed.

3.2 Reciprocal space basis

When parameterizing the field, we use the approximation

$$f(\boldsymbol{r}, \boldsymbol{q}) \approx \boldsymbol{B}_i(\boldsymbol{q}) a_i(\boldsymbol{r}) \tag{3.3}$$

using the Einstein summation convention and where $\boldsymbol{B}$ is a representation for a subset of all functions on the unit sphere, per Sect. 2.3.2. There are several types of representations, which are computed in different ways.

3.2.1 Reconstruction space representation

3.2.1.1 Grid-based representations

In a grid-based representation, the first feature of the representation that must be determined is the grid. Several algorithms of varying complexity for computing spherical grids exist. The simplest grids, such as the rectangular grid with evenly distributed points across polar and azimuthal angles, are not suitable for representation as the points have a very uneven distribution on the sphere. However, such grids have certain other uses, such as when casting a function to spherical harmonics via the Driscoll-Healy sampling theorem [26].

There are several other simple options. One appealing approach is the Kurihara mesh, which divides the sphere into four quadrants and then subdivides those quadrants into triangles, as it is very simple to compute. It can be modified by removing some points in order to decrease its regularity, which is often desirable in order to avoid artefacts from grid points coinciding in different representations. Such a modified Kurihara mesh is the principal grid used in this work.

Another option is the Fibonacci lattice, which evenly divides the sphere by height (cosine of latitude) and takes steps subdivided by the golden mean in the azimuthal direction. There are several useful features of this lattice — an arbitrary number of points can be placed, and except for the areas near the poles it has a fairly regular distribution. This mesh is visually appealing and is used for some of the visualizations in this work.

There are other possibilities, such as those based on icosahedra (which may be further subdivided into triangles to increase density suitable for barycentric interpolation), those based on cubic meshes (which allow easily defined bilinear interpolation to be carried out), and those based on physics-inspired optimization, *e.g.*, finding an electrostatic configuration of N surface charges on a conducting spherical shell. Except for some illustrative purposes, these representations were not used in this work as bilinear and barycentric interpolation were not investigated.

In addition to a grid, grid-based representations require a basis function or an interpolation scheme, and a metric. The most appropriate metric for a Friedel symmetric spherical function is the hemispherical great-circle distance, defined as

$$\Delta(\boldsymbol{r}_1, \boldsymbol{r}_2) = \arccos \frac{|\boldsymbol{r}_1 \cdot \boldsymbol{r}_2|}{\|\boldsymbol{r}_1\| \|\boldsymbol{r}_2\|}. \tag{3.4}$$

Since we generally want our representations to be as directionally unbiased as possible, a suitable type of basis function is the radiaul basis function (RBF), a function of a single non-negative argument. One of the simplest RBFs is the Gaussian, defined simply as

$$G(r) = a \exp\left(-\frac{r^2}{2b}\right)$$

where a and b are amplitude and width parameters, respectively. The Gaussian kernel for each grid point i then becomes

$$G_i\left(\Delta(\boldsymbol{r}_i, \boldsymbol{r})\right) = a \exp\left(-\frac{\Delta(\boldsymbol{r}_i, \boldsymbol{r})^2}{2b}\right)$$

This kernel has many good properties: It is easy and intuitive to configure, it is smooth, and it is relatively localized. Essentially, we want to set b to be large enough that all neighbouring Gaussians are within a full-width half-maximum of each other, so that they pairwise yield unimodal distributions. We can then adjust a on a point-by-point basis to compensate for any lack of uniformity in the distribution of grid points using some simple rule, *e.g.*, that the sum of all basis functions at each grid point should be the same.

There are certain theoretical disadvantages to the Gaussian kernel. Notably, it lacks the property of *compact support*, which essentially means it is not constrained to any one region on the sphere. This issue can be alleviated (if one wishes to obtain a sparse system) by wrapping the Gaussian with a window function (such as a Hamming window), although this introduces additional parameters such as the window width and type. An alternative is to use a kernel which already has compact support. One option with compact support would be the bump function,

$$K_B(r) = \begin{cases} \exp\left(-\frac{1}{1-(\epsilon r)^2}\right) & \text{if } r < \frac{1}{\epsilon} \\ 0 & \text{otherwise} \end{cases}$$

and there are also several families of polynomial splines of r that can be used to similar effect such as Wendland kernels [27]. While it is plausible that in some cases one could obtain improvements by using these alternative kernels, the Gaussian performed sufficiently well for the purposes of this work and these other options have therefore not been implemented.

3.2.1.2 Polynomial representation

Polynomial representation, *i.e.*, expansion in spherical harmonics, is much simpler than grid-based representation. The real-valued spherical harmonics are defined by

$$Y_\ell^m(\theta,\phi) = \begin{cases} L_\ell^m(\cos\theta)\cos(m\phi), \text{ if } m \geq 0, \\ L_\ell^{|m|}(\cos\theta)\sin(|m|\phi), \text{ if } m < 0, \end{cases} \tag{3.5}$$

where L_ℓ^m is the Legendre associated polynomial of degree ℓ and order m, and $0 \leq |m| \leq \ell$. The spherical harmonics are tuned by selecting a band limit (constraining ℓ). A good estimate for the upper bound can be obtained by applying the Nyquist-Shannon sampling theorem to the detector space representation [28], which yields $\ell \leq N - 2$ where N is the number of detector segments after accounting for Friedel symmetry. This is based on the argument that the spherical harmonics projected into a great circle result in a Friedel symmetric trigonometric polynomial in m with 7 degrees of freedom. This argument does not yield an exact constraint, because the fit occurs to integrals rather than to points and measurements from different projection directions can complement each other to some degree. Instead, the result is a rough estimate of a suitable bound.

Spherical harmonics are arguably more useful in analyzing reconstructions than as a basis set themselves. This is primarily because spherical harmonics suffer from the Gibbs phenomenon (also known as ringing artefacts), which causes issues near sharp edges in the reconstruction, and because it is difficult to apply certain types of constraints to a spherical harmonic representation. For example, it is difficult to ensure that a function in a spherical harmonic representation is non-negative, and it is difficult to enforce certain types of regularization (such as L_1 regularization) without breaking the rotational invariance of the spherical harmonics. However, expanding a reconstruction which was carried out using a different representation in spherical harmonics allows for easy computation of various properties such as the mean value, the covariance, and the rank-2 tensor component, from which the main orientation can be calculated. Specifically, the spherical mean is given by a_0^0 and the covariance can be obtained by applying the power-spectral theorem per

$$S_\ell(g,h) = \mathcal{N}(\ell) \sum_{m=-\ell}^{\ell} g_\ell^m h_\ell^m \tag{3.6}$$

$$\mathrm{cov}\,(g,h) = \sum_\ell S_\ell(g,h), \tag{3.7}$$

where S_ℓ is the cross-spectral function of g and h, which are two functions on the sphere expanded in spherical harmonics. $\mathcal{N}(\ell)$ is a normalization term that depends on the choice of the spherical harmonic representation. Using the covariance, we can by definition compute the Pearson correlation coefficient R^2

$$R^2 = \frac{\operatorname{cov}(g,h)^2}{\operatorname{var}(g)\operatorname{var}(h)}, \tag{3.8}$$

where $\operatorname{var}(h) \equiv \operatorname{cov}(h,h)$. This is a very useful quantity for evaluating the similarity of two reconstructions.

Conversion to rank-2 tensor form is carried out by solving the system of equations

$$\begin{bmatrix} a_0^2 & a_1^2 & a_{-1}^2 & a_2^2 & a_{-2}^2 \end{bmatrix} \begin{bmatrix} \frac{2z^2-y^2-x^2}{2\sqrt{3}} \\ xz \\ yz \\ xy \\ \frac{x^2-y^2}{2} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}^\top \begin{bmatrix} T_{xx} & T_{xy} & T_{xz} \\ T_{xy} & T_{yy} & T_{yz} \\ T_{xz} & T_{zy} & T_{zz} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.9}$$

for the rank-2 tensor components T_{ij}. Orientation analysis can then be done by solving the eigenvalue problem for the rank-2 tensor. The main orientation for many symmetries is given by the eigenvector associated with either the smallest or the largest eigenvalue. If the symmetry is unambiguous, this eigenvalue is typically also the largest absolute eigenvalue.

Moreover, filters can be easily applied to a spherical harmonics representation by applying the spherical harmonic convolution theorem, which essentially says that a circularly symmetric convolution kernel can be applied by weighting a spherical harmonic representation according to

$$K(\theta) * (a_\ell^m Y_\ell^m(\theta,\phi)) = S[K]_\ell^0 a_\ell^m Y_\ell^m(\theta,\phi),$$

where we employ the Einstein summation convention for repeated indices, and where K is a convolution kernel that depends only on θ, $*$ is the convolution operator, and S is the spherical harmonic expansion of K. This approach can be used to smoothen reconstructions, but also to compute the Funk-Radon transform [29], which converts a so-called equatorial (or transversal) amplitude into meridional (or longitudinal) amplitude. This is especially useful for visualizing the orientation of tensors with equatorial symmetry.

3.2.2 Detector space representation

When the detector basis functions are integrals along curves (and surface integrals as in Eq. (2.11)), they should be re-written as integrals of scalar variables in order to evaluate them efficiently.

Per the reduction scheme in Sect. 2.3.1, each detector segment will start at an angle φ_1 and end at an angle φ_2. Since the segment lies on a circle on the sphere (a great circle in the small-angle approximation, a small circle more generally) it can be parameterized using three vectors and spherical linear interpolation (SLERP) as

$$\begin{aligned}
\boldsymbol{q}(\varphi) &= \boldsymbol{q}_{\parallel} + \boldsymbol{q}_0 \cos\varphi + \boldsymbol{q}_{90} \sin\varphi, \\
\boldsymbol{q}_1 &= \boldsymbol{q}_0 \cos\varphi_1 + \boldsymbol{q}_{90} \sin\varphi_1, \\
\boldsymbol{q}_2 &= \boldsymbol{q}_0 \cos\varphi_2 + \boldsymbol{q}_{90} \sin\varphi_2, \\
\boldsymbol{q}_D(t) &= \boldsymbol{q}_{\parallel} + \frac{\boldsymbol{q}_1 \sin(t\Delta\varphi) + \boldsymbol{q}_2 \sin((1-t)\Delta\varphi)}{\sin\Delta\varphi},\ t \in [0,1]
\end{aligned}$$

where $\boldsymbol{q}_{\parallel}$ is the component of the reciprocal space vector parallel to the incident beam direction while $\boldsymbol{q}_0$ and $\boldsymbol{q}_{90}$ are two perpendicular components which together with the parallel component parameterize the probed circle in reciprocal space. $\Delta\varphi$ is the difference between φ_1 and φ_2. If this difference is small, for numerical accuracy the approximation $\sin(t\Delta\varphi)\sin\Delta\varphi \approx t$ can be used instead yielding linear interpolation (LERP). When the small-angle approximation is accurate, $\|\boldsymbol{q}_{\parallel}\| \approx 0$.

Writing a general expression for a surface integral is more difficult. A relatively simple option would be to express a mapping of a rectangular kernel by simultaneously rotating both $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ about the axis $\boldsymbol{q}_1 - \boldsymbol{q}_2$, or a component thereof, as a first SLERP and then carrying out a second SLERP at each subdivision over the range of rotation. This would allow the integral to be mapped to the range $[0,1] \times [0,1]$.

If we were to attempt to impose other constraints on the mapping, as described in Sect. 3.2.2, such as

$$\boldsymbol{a}^* = \underset{\boldsymbol{a}}{\operatorname{argmin}} \int_{\hat{l} \in D_h(\hat{l}) \neq 0} \mathrm{d}\hat{l} \left| f_{Dh} \boldsymbol{D}_h\left(\hat{l}\right) - a_i \boldsymbol{B}_i\left(\hat{l}\right) \right|^2 + \Lambda(\boldsymbol{a}) \tag{3.10}$$

where Λ is, *e.g.*, a smoothness constraint, we are faced with a much more difficult problem to tackle. In effect, whether we describe this as a forward minimization problem (optimizing for the mapping onto detector space) or as an inverse minimization problem (optimizing for the mapping onto reconstruction space), we would be potentially forced both to solve an optimization problem and to find an implicit vector-valued function, *e.g.*, $\boldsymbol{f_D}(\boldsymbol{a}, \Lambda)$. This is a rather onerous task and we will therefore not explore this path further but rather leave it as an approach that could be of potential interest in some cases.

3.2.3 Matrix element quadrature

As discussed in Sect. 2.3.2, the most important aspect of mapping between the detector space and the spherical space is to compute the matrix element

$$M_{hk} = \int_{\hat{l} \in D_h \neq 0} \mathrm{d}\hat{l} D_h\left(\hat{l}\right) B_k\left(\hat{l}\right),$$

where D_h is the integration kernel from the sphere to the detector, and B_k is the spherical basis. Using the results of Sect. 3.2.2, we can re-write this as an integral of one variable for the case of a simple curve:

$$M_{hk} = \int_0^1 \mathrm{d}t D_h(\boldsymbol{q}_D(t)) B_k(\boldsymbol{q}_D(t)).$$

This integral can be computed by standard quadrature approaches, such as the adaptive Simpson's rule [30]. The adaptive Simpson's approach, which is summed up in Algorithm 1, involves dividing the integral into subintervals and applying Simpson's rule to each sub-interval until convergence is attained. For very small segments the midpoint rule can be used. This rule is the approximation

$$\int_a^b \mathrm{d}x f(x) \approx (b-a) f\left(\frac{a+b}{2}\right),$$

although cases where this approximation is suitable will generally easily converge with other quadrature approaches.

Cases where simple quadrature rules may fail to converge in a reasonable number of iterations include those where step functions (as in nearest-neighbour interpolation) are used in the basis set due to the Runge phenomenon, which is the quadrature equivalent of the Gibbs phenomenon in signal analysis [31]. In these cases, more complicated methods of numerical integration, such as Gauss-Legendre or Clenshaw-Curtis quadrature, may be more suitable. This may also be the case if the integral is two-dimensional and the integration kernel is not constant, as an uneven distribution of points is likely to improve convergence.

In the event that a more complicated relationship between detector and reconstruction space is used, such as a proximity constraint in addition to another constraint on either representation, the relationship may no longer be a linear one. Moreover, it may be that different constraints are used in the forward and adjoint computation. The reason for this is, as discussed in Sect. 3.1, that we want the reconstruction $\boldsymbol{a} \cdot \boldsymbol{B}$ to approximate the field f, while possibly imposing constraints, not merely for the reconstruction to yield a good approximation of an integral of f. For example, we could use a narrow integration kernel for the forward computation, and a second, more dispersed kernel for the gradient computation. Alternatively, we could impose sparsity conditions on

Algorithm 1 Adaptive Simpson's quadrature for matrix elements

$n \leftarrow 3$
$N \leftarrow 0$
$\epsilon \leftarrow \infty$
while $N < N_{\max}$ and $\epsilon > \epsilon_{\max}$ **do**
  $d \leftarrow 1/(n-1)$
  $i \leftarrow 0$
  $j \leftarrow 0$
  for $i..i_{\max}, j..j_{\max}$ **do**
    $m \leftarrow 0$
    $M_{i,j}^N \leftarrow 0$
    while $m < n-2$ **do**
      $q \leftarrow \frac{d}{3}(B_{ij}(md) + 4B_{ij}((m+1)d) + B_{ij}((m+2)d)$
      $M_{i,j}^N \leftarrow M_{i,j}^N + q$
      $m \leftarrow m+2$
    end while
    if N > 0 **then**
      $\epsilon_{ij} \leftarrow |M_{ij}^N - M_{ij}^{N-1}|$
    else
      $\epsilon_{ij} \leftarrow \infty$
    end if
  end for
  $\epsilon \leftarrow \max_{ij}(\epsilon_{ij}) / \max_{ij}(|M_{ij}^N|)$
  $n \leftarrow 2n-1$
  $N \leftarrow N+1$
end while

the projection of the gradient (such as forcing it to minimize an L_1 norm), which would require not only a projection matrix but also a re-scaling and zeroing of small elements. Because approaches along this line involve potentially complex numerical computations (such as requiring two-dimensional integrals) and because this work is not primarily concerned with sparse data, they have not been implemented. Instead, adaptive Simpson's quadrature over a curve on the unit sphere has been the primary approach used for both the forward and gradient computation.

3.3 John transform quadrature

While the John transform can also be expressed in terms of a projection matrix, it is generally too large to be stored as such in memory. Consider for example an image of

50×50 elements, with 50 projections, mapping to a volume of 50×50×50 elements. Then for each projection angle and pixel, it would be necessary to store an absolute minimum of 50 non-zero matrix elements mapping to the volume, if limited nearest-neighbour interpolation were to be used. This means a total of $50^4 = 6.25 \times 10^6$ non-zero elements, in a sparse matrix of size $50^3 \times 50^3$. This is still a manageable size but considering the additional storage and evaluation penalties of a sparse matrix, the need to store a duplicate for efficient adjoint computation and the fact that the number of elements grows with the fourth power of the system dimension, it quickly becomes apparent that the sparse matrix storage approach is not practical. Thus, it is necessary to carry out the John transform and its adjoint as repeated quadrature operations, and therefore a trade-off between correctness, accuracy and speed is necessary. Good overviews of the technical concerns when implementing tomographic quadrature on a graphics processing unit (GPU) are given in Xu *et al.* (2010) as well as Palenstijn *et al.* (2011), both of which presented important advances in the implementation of such algorithms [32, 33].

A sketch of the tensor tomography version of the algorithm is given in Algorithm 2, where the important modification for tensor tomography is that values from multiple channels, rather than a single scalar value, are integrated over in parallel. See the aforementioned works by Xu *et al.* (2010) and Palenstijn *et al.* (2011) for more detailed discussions of efficiently implementing this algorithm [32, 33]. The approach used is of the so-called slice-bilinear type, with the interpolation algorithm being described in Algorithm 3 (including trilinear interpolation, which is not directly carried out in Algorithm 2 but useful to discuss in understanding why this approach performs well).

We take 5–23 in Algorithm 3 to define the sampling operation $X[\boldsymbol{c}, i]$ on line 11 in Algorithm 2. In Algorithm 3, the trilinear interpolation (lines 17–22) is used if and only if none of the three coordinates in the shifted $\boldsymbol{c}$ is approximately an integer (or is approximately $i + 0.5$ for some integer i in the original, unshifted $\boldsymbol{c}$). Consider disregarding the If-Then-Else branching and directly applying 17–22 when $c_0 = \lfloor c_0 \rfloor = \lceil c_0 \rceil$. Then $w_i = 0$ for $i > 3$, and, *e.g.*, $w_0 = (1 - b)(1 - c)$, and the operation reduces to bilinear interpolation of only 4 points, exactly as if steps 7–11 had been carried out instead. Therefore, if the quadrature points of the John transform are selected such that the interpolation always occurs when one coordinate equals 0.5, the trilinear interpolation which is taken to approximate the underlying field sampled by X will always reduce to bilinear interpolation. This is precisely what Algorithm 2 does by taking steps which begin at 0.5 and are always of length 1 in the slicing direction, and then effectively carrying out trapezoidal quadrature.[1] Taken at an arbitrary angle, a line from one slice to another in a regular, trilinearly interpolated field can trace out

[1] Strictly speaking, the first and last sample should be multiplied by 0.5 for trapezoidal quadrature. The underlying field, however, generally needs to be treated as going to zero near boundaries in order for tomographic theory to be applicable, which renders this concern moot.

Algorithm 2 Sketch of John transform algorithm

Given reconstruction volume $\boldsymbol{X}$ consisting of some number of voxels, each with $i_{\max}$ channels. $\boldsymbol{X}$ is indexed via a set of 3D coordinates $\boldsymbol{c}$, and a voxel index i.
Given a rectangular prism $\boldsymbol{P}$ from $(0,0,0)$ to $(x_{\max}, y_{\max}, z_{\max})$ within which $\boldsymbol{X}$ is defined, structured according to three principal directions $\boldsymbol{s}_0 = (1,0,0)$, $\boldsymbol{s}_1 = (0,1,0)$, $\boldsymbol{s}_2 = (0,0,1)$.
Initialize output projections $\boldsymbol{p}$, with $n_{\max}$ pixels, $m_{\max}$ directions, and $i_{\max}$ channels: $\boldsymbol{p} \leftarrow 0$
for each pixel $n..n_{\max}$ and each direction $m..m_{\max}$ **do**
    Initialize accumulator: $a_i \leftarrow 0$
    Find optimal slicing direction $\hat{\boldsymbol{s}}$ for $\boldsymbol{m}$: $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{s}_{\text{argmax}_k \|\boldsymbol{m}_k\|}$.
    Find 3D coordinates $\boldsymbol{c}$ for pixel n in the plane orthogonal to direction $\boldsymbol{m}$ which cuts through the origin.
    Find offset vector $\boldsymbol{o}$ parallel to direction $\boldsymbol{m}$ such that $\boldsymbol{o} + \boldsymbol{c}$ has coordinate value 0.5 in direction $\hat{\boldsymbol{s}}$, and update: $c \leftarrow c + o$.
    while $\boldsymbol{c}$ in $\boldsymbol{P}$ **do**
        for $i \leftarrow 0$ up to $i_{\max}$ **do**
            Sample reconstruction volume: $a_i \leftarrow a_i + X[\boldsymbol{c}, i]$
        end for
        $\boldsymbol{c} \leftarrow \boldsymbol{c} + \frac{\boldsymbol{m}}{\boldsymbol{m} \cdot \hat{\boldsymbol{s}}}$
    end while
    for $i \leftarrow 0$ up to $i_{\max}$ **do**
        $p_{nmi} \leftarrow \frac{a_i}{|\boldsymbol{m} \cdot \hat{\boldsymbol{s}}|}$
    end for
end for

a polynomial of up to degree 4, since a trilinearly interpolated volume is a first-order function of three independent variables, and a line from one slice to another may cross a boundary in one of the other two directions. In principle, then, taking 3 additional samples per slice would guarantee exact quadrature of the trilinear field, and adding 1 additional central point and applying Simpson's rule should also yield an improvement. However, this is unlikely to improve the outcome of the reconstruction in practice, since the underlying field is assumed to be relatively smooth at the scale of sampling and higher derivatives would therefore be small to begin with. Moreover, there is no reason to expect any systematic bias in over or underestimating the integral, and therefore some cancellation of such errors is likely to occur across the integration range. Thus, such schemes are unlikely to significantly improve the already internally consistent trapezoidal-bilinear quadrature approach, which is consistent with the results reported by Xu and Mueller (2005) [34].

It is useful to briefly discuss the adjoint John transform algorithm, sketched out in

Algorithm 3 Bilinear-trilinear interpolation

1: Given a three-dimensional scalar field $\boldsymbol{V}(x, y, z)$, defined within a rectangular prism of integer side length $\boldsymbol{P}$ from $(0, 0, 0)$ to $(x_{\max}, y_{\max}, z_{\max})$, let the three-dimensional array $X(i, j, k)$ consist of discrete, unit-spaced samples of $\boldsymbol{V}$ from $(0.5, 0.5, 0.5)$ to $(x_{\max} - 0.5, y_{\max} - 0.5, z_{\max} - 0.5)$, indexed by integer values in the prism $\boldsymbol{P'}$, spanning $(0, 0, 0)$ to $(x_{\max} - 1, y_{\max} - 1, z_{\max} - 1)$. For any triplet of integer indices (i, j, k) that does not lie in the prism $\boldsymbol{P'}$, let $X(i, j, k) := 0$.
2: Let $\boldsymbol{c} \equiv (c_0, c_1, c_2)$ be an arbitrary set of coordinates in $\boldsymbol{P}$.
3: Shift $\boldsymbol{c} \leftarrow \boldsymbol{c} - (0.5, 0.5, 0.5)$
4: Let $\lceil r \rceil$ denote the ceiling operation, and $\lfloor r \rfloor$ denote the floor operation.
5: Then let $X[\boldsymbol{c}]$ denote the approximation of $\boldsymbol{V}$ by interpolation of the discrete samples in X according to
6: **if** $c_0 + \delta = \lfloor c_0 + \delta \rfloor$, for some $0 \leq \delta \ll 1$ **then**
7: $c_0 \leftarrow c_0 + \delta$
8: Let $X_0 \leftarrow X\left(\lfloor c_0 \rfloor, \lfloor c_1 \rfloor, \lfloor c_2 \rfloor\right)$, $X_1 \leftarrow X\left(\lfloor c_0 \rfloor, \lfloor c_1 \rfloor, \lceil c_2 \rceil\right)$, $X_2 \leftarrow X\left(\lfloor c_0 \rfloor, \lceil c_1 \rceil, \lfloor c_2 \rfloor\right)$, and $X_3 \leftarrow X\left(\lfloor c_0 \rfloor, \lceil c_1 \rceil, \lceil c_2 \rceil\right)$.
9: Let $a \leftarrow c_1 - \lfloor c_1 \rfloor$, and $b \leftarrow c_2 - \lfloor c_2 \rfloor$
10: Let $w_0 \leftarrow (1 - a)(1 - b)$, $w_1 \leftarrow (1 - a)b$, $w_2 \leftarrow a(1 - b)$, $w_3 \leftarrow ab$
11: Let $X[\boldsymbol{c}] := \sum_{i=0}^{3} X_i w_i$
12: **else if** $c_1 + \delta = \lfloor c_1 + \delta \rfloor$, for some $0 \leq \delta \ll 1$ **then**
13: Do 7–11 but swap operations applied to c_0 and c_1 in 8, and swap c_0 and c_1 in 7 and 9.
14: **else if** $c_2 + \delta = \lfloor c_2 + \delta \rfloor$, for some $0 \leq \delta \ll 1$ **then**
15: Do 7–11 but swap operations applied to c_0 and c_2 in 8, and swap c_0 and c_2 in 7 and 9.
16: **else**
17: Let $X_0 \leftarrow X\left(\lfloor c_0 \rfloor, \lfloor c_1 \rfloor, \lfloor c_2 \rfloor\right)$, $X_1 \leftarrow X\left(\lfloor c_0 \rfloor, \lfloor c_1 \rfloor, \lceil c_2 \rceil\right)$, $X_2 \leftarrow X\left(\lfloor c_0 \rfloor, \lceil c_1 \rceil, \lfloor c_2 \rfloor\right)$, $X_3 \leftarrow X\left(\lfloor c_0 \rfloor, \lceil c_1 \rceil, \lceil c_2 \rceil\right)$
18: Let $X_4 \leftarrow X\left(\lceil c_0 \rceil, \lfloor c_1 \rfloor, \lfloor c_2 \rfloor\right)$, $X_5 \leftarrow X\left(\lceil c_0 \rceil, \lfloor c_1 \rfloor, \lceil c_2 \rceil\right)$, $X_6 \leftarrow X\left(\lceil c_0 \rceil, \lceil c_1 \rceil, \lfloor c_2 \rfloor\right)$, $X_7 \leftarrow X\left(\lceil c_0 \rceil, \lceil c_1 \rceil, \lceil c_2 \rceil\right)$
19: Let $a \leftarrow c_0 - \lfloor c_0 \rfloor$, $b \leftarrow c_1 - \lfloor c_1 \rfloor$, and $d \leftarrow c_2 - \lfloor c_2 \rfloor$.
20: Let $w_0 \leftarrow (1 - a)(1 - b)(1 - d)$, $w_1 \leftarrow (1 - a)(1 - b)d$, $w_2 \leftarrow (1 - a)b(1 - d)$, $w_3 \leftarrow (1 - a)bd$.
21: Let $w_4 \leftarrow a(1 - b)(1 - d)$, $w_5 \leftarrow a(1 - b)d$, $w_6 \leftarrow ab(1 - d)$, $w_7 \leftarrow abc$.
22: Let $X[\boldsymbol{c}] := \sum_{i=0}^{7} X_i w_i$
23: **end if**

Algorithm 4 Sketch of John transform adjoint algorithm

Given measurement space gradient $\boldsymbol{g}$, with $m_{\max}$ directions, some number of pixels, and $i_{\max}$ channels, which is treated as sampling a function spanning the plane $\boldsymbol{P}$ from $(0,0)$ to $(x_{\max}, y_{\max})$, at discrete points from $(0.5, 0.5)$ to $(x_{\max} - 0.5, y_{\max} - 0.5)$.
Initialize volume gradient array $\boldsymbol{G}$, with $n_{\max}$ voxels, and $i_{\max}$ channels: $\boldsymbol{p} \leftarrow 0$
for $n \leftarrow 0$ up to $n_{\max}$ **do**
    for $m \leftarrow 0$ up to $m_{\max}$ **do**
        Initialize accumulator: $a_i \leftarrow 0$
        Find optimal slicing direction $\hat{\boldsymbol{s}}$ for $\boldsymbol{m}$: $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{s}_{\arg\max_k |\boldsymbol{m}_k|}$.
        Find 2D coordinates $\boldsymbol{c}$ for voxel n where a ray in direction $\boldsymbol{m}$ would intersect the projection plane.
        for $i \leftarrow 0$ up to $i_{\max}$ **do**
            Sample projection gradient: $a_i \leftarrow a_i + \frac{g[m,\boldsymbol{c},i]}{|\boldsymbol{m} \cdot \hat{\boldsymbol{s}}|}$
        end for
    end for
    for $i \leftarrow 0$ up to $i_{\max}$ **do**
        $G_{ni} \leftarrow a_i$
    end for
end for

Algorithm 4. This algorithm has an apparent inconsistency in its definition: It does not trace out each ray through the volume, slice-by-slice, and distribute the gradient contribution according to inverse bilinear interpolation of the volume, which would be the naive expectation (it would be the transpose of the implicit matrix of the John transform). Rather, the algorithm bilinearly interpolates a point in projection space which would intersect exactly with the sampled point in volume space, and takes the value of this interpolated point to be the gradient contribution with respect to that measurement direction. This is clearly not the exact adjoint, since the bilinear interpolation is carried out in a different coordinate system and not all points to which each voxel contribute will necessarily contribute to the gradient of that voxel. However, as with the discussion of the sphere-to-detector representation in Sect. 3.2.3, the adjoint does not necessarily need to be the adjoint of the bilinear approximation. Rather, it needs to be an approximation of the continuous adjoint of the John transform, which is an integral of a transformed field that runs over all integration directions, mapping to each point in three-dimensional space. Using bilinear interpolation on a different grid for the adjoint computation provides an advantage, because it ensures that interpolation artefacts in the forward computation are not reflected exactly in the adjoint computation. Thus, Algorithm 2 and Algorithm 4, together with Algorithm 3, summarize the forward and adjoint John transform algorithms and interpolation operations as they are implemented in Mumott.

3.4 Tensor tomography algorithms

The tensor tomography algorithms discussed in this work follow a common structure: Pre-processing to compute reciprocal space projection matrices, weights, and preconditioners, and then utilizing these in combination with an on-demand computed John transform to carry out reconstruction.

Algorithm 5 Sketch of SAXS Tensor Tomography algorithm

1: Load data $\boldsymbol{d}$.
2: Compute all necessary vectors and scalars for carrying out the John transform $\mathbf{P}[\cdot]$.
3: Compute reciprocal space projection matrix $\boldsymbol{M}$.
4: Compute weights $\mathbf{w}$ and preconditioning matrix $\mathbf{W}$. Let $\mathbf{W}$ encode the step size for each update of the solution. In the simplest case, $\mathbf{w}$ is the identity matrix, and $\mathbf{W}$ is the identity matrix multiplied by a constant step size.
5: Solution $\mathbf{X} \leftarrow \mathbf{0}$.
6: **while** $i < i_{\max}$ or until convergence **do**
7: Compute John transform of solution $\boldsymbol{p} \leftarrow \mathbf{P}[\mathbf{X}]$
8: Project solution into detector space $\boldsymbol{p}' \leftarrow \boldsymbol{p}\mathbf{M}$
9: Compute the gradient of the weighted residual norm with respect to the projected solution, *e.g.*, $\boldsymbol{g} \leftarrow \mathbf{w}(\boldsymbol{p}' - \boldsymbol{d})$), and if used to determine convergence, compute the residual norm itself and add to loss function, *e.g.*, $L \leftarrow \frac{1}{2}(\boldsymbol{p}' - \boldsymbol{d})^T\mathbf{w}(\boldsymbol{p}' - \boldsymbol{d})$.
10: Compute the gradient of the residual with respect to the projection from reconstruction reciprocal space into data space, $\boldsymbol{g}' \leftarrow \mathbf{M}\boldsymbol{g}$.
11: Compute the adjoint of the John transform of the partial gradient $\boldsymbol{g}'$, weighted by the preconditioning matrix: $\boldsymbol{G} \leftarrow \mathbf{W}P^T[\boldsymbol{g}']$.
12: Compute the gradients with respect to any regularization norms, and if used, the regularization norms. Add to gradient and loss function respectively: $\boldsymbol{G} \leftarrow \boldsymbol{G} + \nabla_{\boldsymbol{X}}\Lambda(\boldsymbol{X})$, and $L \leftarrow L + \Lambda(\boldsymbol{X})$.
13: Update solution: $\boldsymbol{X} \leftarrow \boldsymbol{X} - \boldsymbol{G}$
14: Compute *e.g.*, change in loss function or magnitude of gradient, if used to determine convergence.
15: $i \leftarrow i + 1$
16: **end while**

In Algorithm 5, an overview of a gradient descent-based algorithm is given. This algorithm uses the squared loss function, but in principle more general methods can be used. Moreover, with only minor modifications (such as an additional gradient storage), momentum-based and quasi-Newton methods suitable for large-scale optimization problems can be used, such as Nestorov gradient descent, the conjugate gradient method, and LBFGS.

The separation of the John transform and the RSMping can be emphasized by writing the residual calculation in the form

$$r_h(s,j,k) = \text{RSM}_h(s,j,k) - \mathbf{M}_{hi} \int_{-\infty}^{\infty} \mathrm{d}t\, a_i \left(\boldsymbol{v}(s,j,k) + t\boldsymbol{p}(s)\right),$$

where Einstein summation over repeated indices is used. It can be understood from this formulation that we are able to compute the summation over i for this step and the summation over j necessary in the gradient computation separately from the John transform quadrature, as specified in Algorithm 5.

A potentially interesting opportunity for optimization occurs when the matrix $\boldsymbol{M}$ is sparse, such as when using small detector segments and a high-resolution RSM basis. In this case, rather than converting the projection of the solution into detector space, or the loss function gradient into reconstruction space, an appealing option is to define a modified John transform, which we may represent by moving the matrix $\boldsymbol{M}_{ij}$ inside the integral sign, resulting in

$$r_h(s,j,k) = \text{RSM}_h(s,j,k) - \int_{-\infty}^{\infty} \mathrm{d}t\, \mathbf{M}_{hi} a_i \left(\boldsymbol{v}(s,j,k) + t\boldsymbol{p}(s)\right)$$

The corresponding modification in implementation is reflected in Algorithm 6. The inner loop over all reconstruction channels up to $i_{\max}$ is replaced with a double loop. The outer loop goes over all detector segments up to $j_{\max}$. The inner loop goes over all non-zero elements in the sparse projection matrix for that particular detector segment, the sparse indices of which are stored in the pointer array, and the non-sparse indices and values of which are stored in the index and value arrays.

There are three main advantages of this approach. First, if each detector segment only maps to one, or a few, reconstruction basis elements and there are many more reconstruction basis elements than detector segments, the overall penalty for the extra searches and extra arithmetic operation may be relatively small, when compared to the cost of iterating over all reconstruction basis channels in a much larger projection matrix. Second, there is no need to allocate memory for storing projections with the reconstruction reciprocal space basis and the memory can be re-used for the residual. Third, there is no longer a need to carry out a separate matrix multiplication for every pixel. Consequently, Algorithm 6 enables a very lean and efficient implementation of asynchronous computation for the subset of cases where the assumptions of sparsity are applicable.

The adjoint can be defined by modifying Algorithm 4 in a similar way. The asynchronous solution algorithm which follows by modifying Algorithm 5 is given in Algorithm 7.

Since the value of the loss function is not used for a fully asynchronous computation, optimizations such as pre-weighting the data array and computing the residual using a

Algorithm 6 Sketch of sparse John transform algorithm

As 1 and 2 in Algorithm 2.
Given also a set of sparse projection matrices M_{mji} for $m_{\max}$ directions, $i_{\max}$ channels, and $j_{\max}$ detector segments per direction.
Compute compressed sparse row (CSR) representation of M_{mji}, giving pointer array M^{ptr}_{mj}, index array M^{ind}_{ml}, and value array M^{val}_{ml}.
Initialize output projections $\boldsymbol{p}$, with $n_{\max}$ pixels, $m_{\max}$ directions, and $j_{\max}$ channels: $\boldsymbol{p} \leftarrow 0$
for each pixel $n \leftarrow 0$ up to $n_{\max}$, and each direction $m \leftarrow 0$ up to $m_{\max}$ **do**
    Initialize accumulator: $a_j \leftarrow 0$
    As 6 to 8 in Algorithm 2
    while $\boldsymbol{c}$ in P **do**
        for $j \leftarrow 0$ up to $j_{\max}$ **do**
            for $l \leftarrow M^{\text{ptr}}_{mj}$ up to $M^{\text{ptr}}_{m(j+1)}$ **do**
                Sample reconstruction volume: $a_j \leftarrow a_j + X[c, M^{\text{ind}}_{ml}]M^{\text{val}}_{ml}$
            end for
        end for
        $\boldsymbol{c} \leftarrow \boldsymbol{c} + \frac{\boldsymbol{m}}{\boldsymbol{m}\cdot\hat{\boldsymbol{s}}}$
    end while
    for $j \leftarrow 0$ up to $j_{\max}$ **do**
        $p_{nmj} \leftarrow \frac{a_j}{|\boldsymbol{m}\cdot\hat{\boldsymbol{s}}|}$
    end for
end for

fused multiply-add operation can be employed. If implemented correctly, the computation kernels in Algorithm 7 can be computed and sent to the GPU well in advance of the actual computations being carried out, reducing overhead. It is also possible to synchronize only occasionally, check convergence, and continue the computations if convergence has not been achieved yet. For example, the solution and forward projection could be copied to standard RAM every 10 iterations and the loss function could then be computed using CPU resources after the instructions for the next 10 iterations have been sent to the GPU. Alternatively, the change in the loss function could be computed directly using GPU resources, but only transferred to regular RAM and checked once every 10 or so iterations. This is in stark contrast to a standard mixed computation implementation following Algorithm 5, where only the John transform and the adjoint would be computed on the GPU, requiring two separate copying operations to GPU memory and two separate copying operations back to the CPU for every iteration.

It is also relatively easy to implement a similar albeit somewhat more memory-demanding version of asynchronous reconstruction for non-sparse cases. This is because

Algorithm 7 Sketch of asynchronous, sparse SAXSTT algorithm

1: Load data $\boldsymbol{d}$.
2: Compute all necessary vectors and scalars for carrying out the John transform $\mathcal{P}[\cdot]$.
3: Compute reciprocal space projection matrix $\boldsymbol{M}$, and its sparse representation, denoted $\boldsymbol{M}_{\text{csr}}$.
4: Compute weights $\mathbf{w}$ and preconditioning matrix $\mathbf{W}$, and choose a step size t.
5: Allocate solution array $\boldsymbol{X} \leftarrow 0$.
6: Allocate gradient array $\boldsymbol{G} \leftarrow 0$.
7: Allocate projection array $\boldsymbol{p} \leftarrow 0$
8: Move all arrays to be used to the GPU.
9: **while** $i < i_{\text{max}}$ **do**
10: Compute sparse-matrix John transform of solution $\boldsymbol{p} \leftarrow \mathcal{P}[\boldsymbol{X}, \mathbf{M}_{\text{csr}}]$
11: Compute the gradient of the weighted residual norm with respect to the projected solution and store in the same array as the projection, *e.g.*, $\boldsymbol{p} \leftarrow \mathbf{w}(\boldsymbol{p} - \boldsymbol{d})$.
12: Compute the adjoint of the sparse-matrix John transform of the partial gradient $\boldsymbol{g}$, weighted by the preconditioning matrix: $\boldsymbol{G} \leftarrow \mathbf{W}\mathcal{P}^T[\boldsymbol{g}, \boldsymbol{M}_{\text{csr}}]$.
13: Compute the gradients with respect to any regularization norms. Add to gradient: $\boldsymbol{G} \leftarrow \boldsymbol{G} + \nabla_{\boldsymbol{X}}\Lambda(\boldsymbol{X})$.
14: Update solution: $\boldsymbol{X} \leftarrow \boldsymbol{X} - \boldsymbol{G}$
15: $i \leftarrow i + 1$
16: **end while**

the contraction operations in Algorithm 5 can be represented as a series of matrix-vector products instead of matrix-matrix products. In particular, the two contraction operations are

$$a'_{sjkh} = \boldsymbol{M}_{shi} a_{sjki},$$
$$r'_{sjki} = \boldsymbol{M}_{shi} r_{sjkh},$$

where we note that contraction only occurs over the last index in a and r, respectively. While it would be possible to structure these operations as matrix-matrix products, doing so efficiently would require re-structuring of the data. Since this contraction operation is the less demanding operation when compared to the John transform, it does not seem sensible to structure the data around this computation. Thus, carrying out this computation as a series of matrix-vector products appears sensible and sufficiently efficient. Matrix-vector products are substantially easier to optimize than matrix-matrix products as their memory-access patterns are far simpler. It is also straightforward to implement each of these operations separately as a CUDA kernel. This makes it relatively easy to write a dense-matrix SAXSTT implementation.

The dense-matrix asynchronous approach is sketched out in Algorithm 8. It is fairly similar to the sparse-matrix approach of Algorithm 7, but requires an additional storage

Algorithm 8 Sketch of general asynchronous SAXSTT algorithm

1: Load data $\boldsymbol{d}$.
2: Compute all necessary vectors and scalars for carrying out the John transform $\mathcal{P}[\cdot]$.
3: Compute reciprocal space projection matrix $\mathbf{M}$
4: Compute weights $\mathbf{w}$ and preconditioning matrix $\mathbf{W}$, choose step size t.
5: Allocate solution array $\mathbf{X} \leftarrow \mathbf{0}$.
6: Allocate gradient array $\mathbf{G} \leftarrow \mathbf{0}$.
7: Allocate projection array $\mathbf{p} \leftarrow \mathbf{0}$
8: Allocate reconstruction tensor projection array $\boldsymbol{P} \leftarrow \mathbf{0}$
9: Move all arrays to be used to the GPU.
10: **while** $i < i_{\max}$ **do**
11: Compute John transform of solution $\boldsymbol{P} \leftarrow \mathcal{P}[\boldsymbol{X}]$
12: Project John transform of solution to detector space $p_{sjkh} \leftarrow M_{shi} P_{sjki}$
13: Compute the gradient of the weighted residual norm with respect to the projected solution and store in the same array as the projection, *i.e.*, $\boldsymbol{p} \leftarrow \mathbf{w}\,(\boldsymbol{p} - \boldsymbol{d})$.
14: Project gradient back into solution space, store in $\boldsymbol{P}$: $P_{sjki} \leftarrow M_{shi} p_{sjkh}$
15: Compute the adjoint of the John transform of the partial gradient stored in $\boldsymbol{P}$, weighted by the preconditioning matrix: $\boldsymbol{G} \leftarrow \mathbf{W}\mathcal{P}^T[\boldsymbol{P}]$.
16: Compute the gradients with respect to any regularization norms. Add to gradient: $\boldsymbol{G} \leftarrow \boldsymbol{G} + \nabla_{\boldsymbol{X}} \Lambda(\boldsymbol{X})$.
17: Re-scale gradient by step size t: $\boldsymbol{G} \leftarrow t\boldsymbol{G}$
18: Update solution: $\boldsymbol{X} \leftarrow \boldsymbol{X} - \boldsymbol{G}$
19: $i \leftarrow i + 1$
20: **end while**

array as well as separate John Transform and matrix multiplication operations. It would be possible to also combine the John Transform and the dense matrix in order to reduce memory usage, but this is not necessarily efficient, and requires more code duplication to be carried out.

3.5 Loss functions

The loss function of an optimization algorithm is not always explicitly computed, however, it is generally implied by the choice of update term. Most of the discussion thus far has focused on the weighted squared loss, which has the form

$$\mathcal{L}_2(\boldsymbol{x}, \boldsymbol{d}) = (\boldsymbol{x} - \boldsymbol{d})^T \mathbf{w} (\boldsymbol{x} - \boldsymbol{d}), \tag{3.11}$$

for projection $\boldsymbol{x}$, data $\boldsymbol{d}$, and matrix of weights $\mathbf{w}$. In the trivial case, the weight matrix is simply the identity matrix such that Eq. (3.11) reduces to ordinary least squares. In

general, the weight matrix can encode geometric information as in the simultaneous iterative reconstruction technique (SIRT) case, but it can also be used to encode information about the measurement uncertainty, which is discussed in Sect. 3.5.1.

The other principal loss function of interest is the *Huber loss*, which splices the L_1 and L_2 norms with a parameter δ. Minimizing this type of norm is a type of robust regression and one version of the norm may be written

$$\mathcal{L}_H(\boldsymbol{x}, \boldsymbol{d}, \delta) = \sum_i \begin{cases} \frac{1}{2\delta}(x_i - d_i)^T w_{ij}(x_j - d_j) & \text{if} |x_j - d_j| < \delta \\ |w_{ij}(x_j - d_j)| - \frac{\delta}{2} & \text{otherwise.} \end{cases} \tag{3.12}$$

Depending on the choice of δ, one can balance the robustness of the reconstruction to outliers against the ease of solving the system. It is generally not very useful to use uncertainty-based weights with this type of loss function, as a single large deviation is not privileged over many small ones, although there is some potentially useful information such as the large uncertainty of areas where the transmission is small. The use of such weights need to be balanced against the potential issue of making it more difficult to find the minimum.

Geometry-based weights can be used more freely, but it is still necessary to pick a step size based on the magnitude of the data. Using SIRT weights here can be helpful to eliminate geometry-based influences on the step size. Solving a least-squares problem is often around an order of magnitude faster than solving a robust regression problem, if statistical weights are not used. However, robust regression combined with a total variation regularization is a very reliable approach for reconstruction.

3.5.1 Weights and preconditioners

3.5.1.1 Tensor SIRT

An interesting approach, which can be written in terms of weights and preconditioners, is SIRT, a classic tomographic algorithm. Essentially, SIRT is based on normalizing the residual of a tomography problem by the path length of each line integral within the volume considered, and then normalizing the gradient based on the number of rays that contribute to each point in the adjoint. In other words,

$$\mathbf{w} = \text{diag}\left(\mathcal{P}[1]\right),$$

where $\mathcal{P}$ is the John transform and 1 is a solution vector filled with the value 1 everywhere, and

$$\mathbf{W} = \text{diag}\left(\mathcal{P}^T[1']\right),$$

where $\mathcal{P}^T$ is the adjoint of the John transform and $1'$ is a projection-space vector filled with the value 1 everywhere. This has the effect of normalizing the reconstruction

such that step size unity can be used for gradient descent, and it tends to speed up reconstruction and make it easier to obtain a smooth reconstruction, as all parts of the solution converge at a similar rate.

It is possible to directly apply this to tensor tomography, by simply applying the same weights and preconditioners to all basis functions in each representation, considering only the real-space part of the problem. But the approach can be further extended to a problem with a tensor representation by choosing

$$\mathbf{w} = \mathrm{diag}\left(\mathcal{P}[1]\mathbf{M}\right)$$

for the weights, where $\mathbf{M}$ is the detector-to-sphere mapping matrix, and

$$\mathbf{W} = \mathrm{diag}\left(\mathbf{M}\mathcal{P}^T[1']\right)$$

for the preconditioner. This will tend to smoothen out, *e.g.*, biases in the gradient due to the choice of representation. The drawback is that the use of weights means that the solution obtained via SIRT differs from the solution obtained via ordinary least squares.

3.5.1.2 Noise estimation

If a detector registers an average count of I across n pixels, and the transmission for this measurement is T, then the entry in the data vector will be I/T. Using Poisson statistics and the central limit theorem, we can estimate the uncertainty of this measurement. A single measurement of I has variance I, and per the central limit theorem, averaging over n pixels decreases the standard error by a factor of about $\sqrt{n}$. In other words, the variance of the averaged measurement can be estimated to be I/n. Accounting also for the transmission correction, we end up with a final estimate for our variance as I/nT^2. Encoding the inverse of this value into $\boldsymbol{w}$ will scale the gradient in a way that accounts for this uncertainty and ensures that values with a smaller uncertainty are prioritized in the optimization.

If we also have an estimate for the uncertainty in the transmission, we can carry out a more complicated error propagation given by

$$d = \frac{I \pm \sigma_I}{T \pm \sigma_T} \approx \frac{I}{T} \pm \frac{I\sigma_T + T\sigma_I}{T^2}. \tag{3.13}$$

This expression for the uncertainty of a data point is especially useful as it encodes the fact that uncertainties in the transmission become very important when the transmission is small. This can counteract influence by outliers that may occur in low-transmission areas by encoding this information into the weights of the loss function.

3.5.1.3 Iteratively reweighted norm

One promising use of weights and preconditioners which can be approached in a few different ways is the iteratively reweighted norm (IRN) approach, which is based on the iteratively reweighted least squares (IRLS) method [35]. In this method, one solves a problem using a diagonal weight matrix that gradually transforms a least-squares problem into a p-norm problem. In a simple case, we transform the problem of solving for the smallest p-norm of a residual,regularized by minimizing the t-norm of the solution vector,into a weighted least squares problem. This least-squares problem has the loss function

$$\begin{aligned} \mathcal{L}_{pt} &= \|\boldsymbol{x} - \boldsymbol{d}\|_p^p + \lambda \|\boldsymbol{X}\|_t^t \\ &= \left\|\mathbf{w}_p (\boldsymbol{x} - \boldsymbol{d})\right\|_2^2 + \lambda \left\|\mathbf{W}_t \boldsymbol{X}\right\|_2^{2\cdot} \end{aligned} \tag{3.14}$$

Here, we initially set $\mathbf{W}_p^0$ and $\mathbf{w}_p^0$ to the identity matrix and then refine them iteratively, clamped by a Huber approximation for small values,

$$\begin{aligned} \mathbf{w}_p^i &= \left(\max(\boldsymbol{x}^{\boldsymbol{i}-1} - \boldsymbol{d}, \epsilon_p)\right)_p^{p-2} \\ \mathbf{W}_t^i &= \left(\max(\boldsymbol{X}^{\boldsymbol{i}-1}, \epsilon_t)\right)_t^{t-2} \end{aligned},$$

where $\boldsymbol{X}^i$ is obtained by solving

$$\boldsymbol{X}^i = \underset{\boldsymbol{X}}{\operatorname{argmin}} \left(\left\|\mathbf{w}_p^i (\boldsymbol{x} - \boldsymbol{d})\right\|_2^2 + \lambda \left\|\mathbf{W}_t^i \boldsymbol{X}\right\|_2^2 \right).$$

As the number of iterations increases, $\boldsymbol{X}^i$ converges towards a Huber-regularized solution that minimizes Eq. (3.14) with respect to $\boldsymbol{X}$.

This approach can be simpler than the Huber approach, since it is typically not necessary to do a step-size search as the step size for the least-squares problem can be directly estimated. It is also not necessary to redefine the computation of the gradient for every value of p and t, which is especially useful for more complicated regularization terms.

3.5.2 Tensor tomography regularization

In principle, tensor tomography regularization does not need to be particularly different from scalar tomography regularization. The main complication is the question of how to deal with representation-specific concerns. There are three main concerns: convergence rate, reconstruction smoothness, and converging without overfitting to the data. Regularization helps with these issues by imposing additional constraints on the reconstruction. In general we can identify four types of constraints:

- Point-by-point norm regularization such as constraining the L_1 or (squared) L_2 norm of the reconstruction. This type of regularization considers every basis function in every voxel individually. This is the simplest type of regularization. It can help to reduce background noise as well as speeding up convergence.

- Voxel-by-voxel norm regularization. This type of regularization considers each voxel individually, and does not necessarily decouple the basis functions but rather considers the spherical function as a whole. Certain norms (*e.g.*, the variance) may belong to the point-by-point type for some basis sets (such as spherical harmonics) and the voxel-by-voxel type for others (*e.g.*, most local representations would need to undergo a transforma, such as into spherical harmonic space, in order to determine the variance). An example of a voxel-by-voxel norm would be the Euclidean norm of the basis functions of each voxel. Another example would be the Euclidean norm for each frequency band in a spherical harmonic representation.

- Real-space variation regularization, which considers the relationship between tensors in nearby voxels and imposes a constraint on some measure of this relationship, such as the Laplacian or the total variation, between voxels. Generally this type of regularization is used to make the solution more smooth, which also helps with the convergence rate and the avoidance of overfitting.

- Reciprocal-space variation regularization, where one considers how similar basis functions are that are close to each other on the reciprocal space sphere. This type of regularization could be of interest for sparsely sampled data. However, similar effects to applying this kind of regularization can be obtained through a suitable choice of reciprocal space representation and segment-to-reciprocal space sphere mapping. Therefore this type of regularization will not be considered in detail.

Point-by-point regularization is appealing largely due to its simplicity. Essentially, one includes a simple norm of the reconstruction (such as the sum of all absolute coefficient values for L_1 regularization or the sum of all squared coefficient values for L_2 regularization) and obtains a predictable effect — the reduction of small values and background noise for L_1 regularization and rapid convergence as well as the damping of large values for the L_2 norm. It can be difficult to get L_1 regularization to converge since the gradient is not smooth, and for this reason it is often useful to employ a smoothened version of the norm such as the L_1 norm spliced with an L_2 norm for small values. This is analogous to the Huber loss function in Eq. (3.12).

Voxel-by-voxel norm regularization has similar aims as point-by-point regularization. It allows regularization to be applied to a region of space, rather than to

individual points in RSMs. This can be advantageous especially when accounting for the difficulty of constraining some parts of reciprocal space. By considering an entire voxel, additional information from other parts of the reciprocal space can be used to constrain the underdetermined parts. For example, when using the Euclidean norm of the entire spherical function as a regularization norm, the entire norm is considered and if well-determined parts of the RSM are very small, this can constrain less well-determined parts of the RSM to be small as well.

Real-space variation regularization encourages smoothness of the reconstruction. This is arguably the most important type of regularization, as it dampens fluctuations in the reconstruction and encourages continuity. This helps prevent overfitting to noisy data and its usage reflects real physical knowledge about the system since generally experimental parameters such as beam and step size should be chosen such that the reconstruction can be expected to be reasonably smooth. Two options of interest are the total variation between neighbours (the absolute value of the real-space gradient) as well as the Laplacian (the real-space second derivative). The Laplacian, approximated by the second-order finite difference, is especially interesting in the context of spherical harmonics, as minimizing it corresponds to maximizing the covariance between neighbours, and it is easy to minimize, *e.g.*, the squared norm of the Laplacian. However, it can excessively smoothen the solution and for this reason, total variation is often preferable. Like the L_1 norm, the total variation can be made to converge more easily by splicing it with the Laplacian at small variations.

The total variation can also be extended to the case of entire voxels. The scalar, one-sided[2] total variation can be defined as

$$\mathrm{TV}(\boldsymbol{X}) = \left\| \frac{1}{2} \sqrt{\sum_{i=0}^{2} (\boldsymbol{D}_i \boldsymbol{X})^2} \right\|_1^1, \tag{3.15}$$

with $\boldsymbol{D}_i$ being the finite difference operator in the forward direction for the Cartesian coordinate i. For example for $i = 0$

$$[\boldsymbol{D}_0 \boldsymbol{X}](x, y, z) = X(x, y, z) - X(x + 1, y, z).$$

Equation (3.15) can be applied directly to tensor tomography by simply computing this quantity for each basis function separately. But it is also possible to consider the total variation over all the basis functions at once, that is,

$$\mathrm{VTV}(\boldsymbol{X}) = \left\| \frac{1}{2} \sqrt{\sum_{h} \sum_{i=0}^{2} \left(\boldsymbol{D}_i \boldsymbol{X}_h \right)^2} \right\|_1^1, \tag{3.16}$$

[2]It is most common to use the one-sided total variation, which minimizes the variation *between* grid points, rather than *at* grid points.

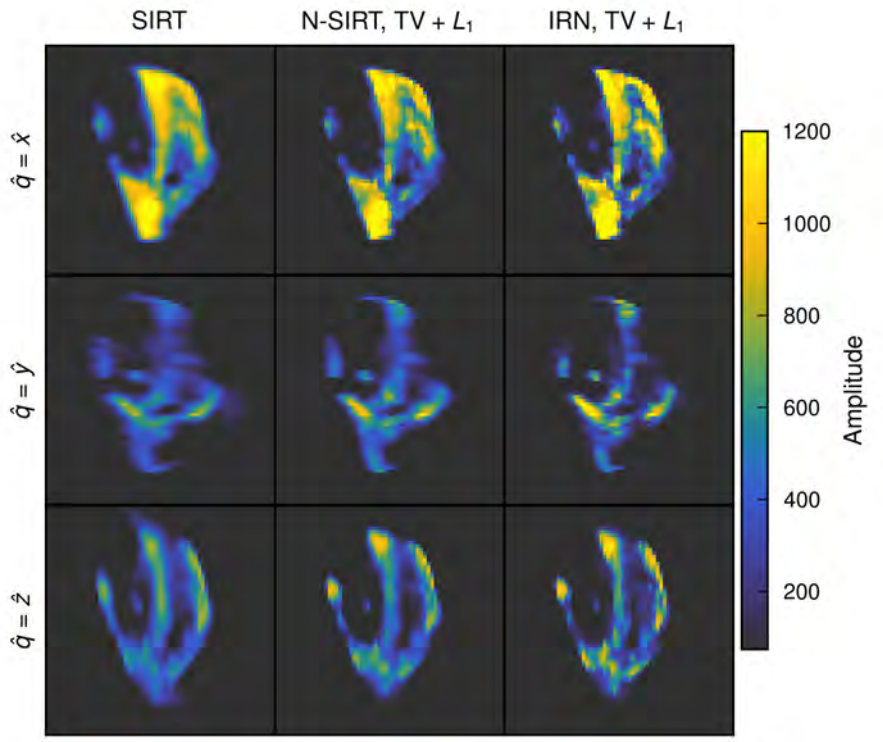

Figure 3.1: Comparison between three reconstructions for a trabecular bone sample, looking at three components of the RSM. Leftmost column: Tensor SIRT with no regularization and 50 iterations. Middle column: Nesterov-accelerated SIRT with total variation and Huber norm regularization. Rightmost column: IRN least-squares reconstruction with voxel-wise total variation and coefficient-wise L_1 regularization.

where each h indexes a basis function. We can combine this approach with the IRLS detailed in Sect. 3.5.1, by calculating

$$\mathrm{VTV}\left(\boldsymbol{X}_i\right) \approx \frac{\mathrm{VTV}\left(\boldsymbol{X}_i\right)^2}{\mathrm{VTV}\left(\boldsymbol{X}_{i-1}\right)},$$

where $\boldsymbol{X}_i$ is the reconstruction currently being optimized and $\boldsymbol{X}_{i-1}$ is the reconstruction of the previous optimization. A similar approach can be used to optimize for, *e.g.*, the Euclidean norm of the coefficient vector in each voxel, which can be useful in thresholding reconstructions.

We can see a few different approaches to regularization applied to trabecular bone in Fig. 3.1, which has $\hat{\boldsymbol{x}}$ as its main tomographic axis. The tensor SIRT reconstruction is simple and leads to a smooth reconstruction, but it is blurry and suffers from missing wedge artefacts, especially the y and z-components. On the other hand, the Nesterov-accelerated SIRT reconstruction that is total variation and Huber norm regularized

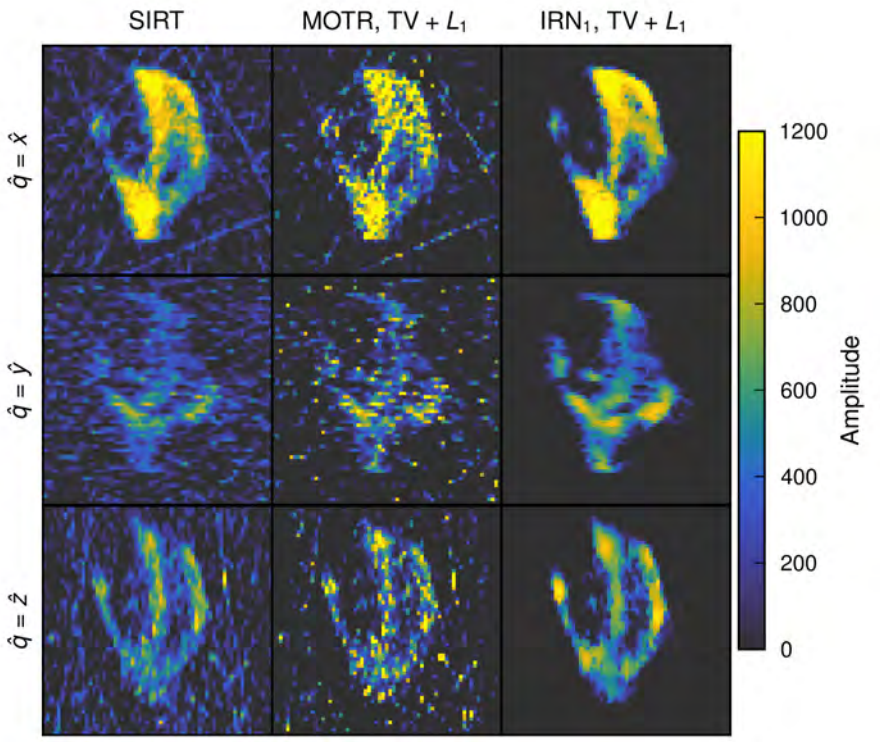

Figure 3.2: Comparison between three reconstructions for a trabecular bone sample with large amounts of added noise, looking at three components of the RSM. Leftmost column: Tensor SIRT with no regularization and 50 iterations. Middle column: Nesterov-accelerated SIRT with total variation and Huber norm regularization. Rightmost column: IRN L_1 reconstruction with voxel-wise total variation and coefficient-wise L_1 regularization.

performs better, reducing the smearing, since the regularization enforces sparsity. Finally, consider the IRN reconstruction which enforces voxel-wise total variation and coefficient-wise L_1 regularization. Both types of regularization appear to be enforced more effectively with this reconstruction algorithm, making it possible to push reciprocal space sparsity and continuity to relatively extreme lengths. The reconstruction has well-defined as well as larger, more localized amplitudes for the $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{z}}$ components of the RSM.

A similar comparison is shown in Fig. 3.2, where large amounts of noise has been added to the reconstruction, and an L_1 norm is used for the IRN residual, making this a robust regression reconstruction [36]. SIRT and momentum total variation reconstruction (MOTR) both suffer heavily from the noise, resulting in many artefacts, which are especially strong in MOTR. This is likely because MOTR converges more quickly, and is therefore be more prone to overfitting. However, IRN with an L_1 residual norm is relatively robust against the noise; while the reconstruction quality does not appear to

be as good as any example in Fig. 3.1, there are no noise artefacts being reconstructed in empty space, and the reconstruction is serviceable. This robustness to outliers is a major advantage of robust regression.

3.6 Other approaches

There are optimization approaches which are similar but not identical to the gradient-based approach. Here, we will consider *proximal methods* as well as *direct reconstruction*.

3.6.1 Proximal methods

Rather than using gradients, it is possible to use *proximal operators* for optimization. Applying a proximal operator is similar to updating a solution with a gradient but more general. In principle, each application of a proximal operator is in itself an optimization problem in so far as one tries to find the *most proximal* vector to the current estimate, which fulfills some additional condition. Proximal operators can exhibit some behaviours that are not possible to precisely obtain using only gradients. The proximal operator is defined by

$$\text{prox}_f(x) = \text{argmin}_z \left[f(z) + \frac{1}{2} \|x - z\|_2^2 \right].$$

For example, let $f(x)$ be the L_1 norm scaled by a weight λ. In this case, it can be shown that

$$\text{prox}_{\lambda\|\cdot\|_1}(\boldsymbol{x}) = \text{sgn}\left(x_i\right) \max\left(\left|x_i\right| - \lambda, 0\right) \forall x_i \in \boldsymbol{x} \tag{3.17}$$

Here, where a gradient-based approach would just subtract $\lambda\,\text{sgn}(x)$, the proximal operator subtracts this term and also sets to zero any coefficients that would otherwise flip the sign. This means that noise terms smaller than λ in the solution will simply be set to 0, counteracting oscillation between small positive and negative values on every iteration. Similarly, for the L_2 norm,

$$\text{prox}_{\lambda\|\cdot\|_2}(\boldsymbol{x}) = \max\left(1 - \frac{\lambda}{\|\boldsymbol{x}\|_2^2}, 0\right) x. \tag{3.18}$$

These operations are used in the *proximal gradient* algorithm, which is of particular interest when one wishes to solve a differentiable problem (such as the tomography problem) subject to a constraint expressed by a function which is not differentiable everywhere, such as the L_1 norm.

Algorithm 9 sketches the proximal gradient algorithm. The algorithm is in practice very similar to the general gradient-based approach (Algorithm 5) with the difference

Algorithm 9 Proximal gradient algorithm for SAXSTT

1: Let $F(\mathbf{X}) := H(\mathbf{X}) + G(\mathbf{X})$. The objective is to minimize $F(\mathbf{X})$.
2: Let $H(\mathbf{X})$ be a differentiable loss function (e.g., the squared loss or Huber loss) of the John transform of the reconstruction $\mathbf{X}$, projected into detector space, and the measured data $\mathbf{d}$.
3: Let $G(\mathbf{X})$ be a possibly non-differentiable norm, such as the L_1 norm.
4: **for** $i \leftarrow 0$ up to $i_{\max}$ **do**
5: Update solution $\mathbf{X}$ with step size ω: $\mathbf{X} \leftarrow \mathbf{X} - \omega \nabla H(\mathbf{X})$.
6: Compute proximal operator of updated solution with respect to G: $X \leftarrow \text{prox}_{\mathbf{G}}(\mathbf{X})$.
7: **end for**

that the gradient of the regularization function is replaced with a proximal operator. In practice, the usefulness of Algorithm 9 depends on the proximal operator being easy to evaluate, which is primarily the case for so-called *simple* proximal operators, i.e., those that have a closed-form expression. Other proximal algorithms exist which can handle non-differentiable loss functions, such as the Chambolle-Pock algorithm. These are generally significantly more complicated to employ than gradient-based algorithms and require a very large number of iterations, but are guaranteed to converge. Several tomography algorithms based on the Chambolle-Pock algorithm have been derived by, *e.g.*, Sidky *et al.* (2010) [37], which could in principle be applied to tensor tomography by incorporating the reciprocal space projection into the system matrix. However, the examples shown by Sidky *et al.* required thousands to tens of thousands of iterations to converge, although it is possible that appropriate preconditioning of the problem could improve this. Therefore these algorithms are likely to be useful mainly for specialized applications in tensor tomography, such as reconstruction from very noisy images. In view of this, the IRLS-weighted approach sketched out in Sect. 3.5.1 appears more promising.

3.6.2 Direct reconstruction

It is interesting to consider the potential of applying some analogy of the inverse Radon formula to tensor tomography. The inverse Radon formula, or filtered back-projection (FBP), can easily be derived from the projection-slice theorem. Somewhat more intuitively, one can understand it by considering the operation of applying the adjoint of the Radon transform to measured data, that is,

$$\mathcal{R}^T[\mathbf{d}] = \mathcal{R}^T[\mathcal{R}[\boldsymbol{X}]] := \mathbf{A}\boldsymbol{X} \tag{3.19}$$

The operator $\mathbf{A}$ is essentially a convolution which exaggerates low spatial frequencies relative to high spatial frequencies in the plane orthogonal to the axis of projection.

Hence, the original image $\boldsymbol{X}$ can be retrieved by deconvolving with a high-pass filter, such as the *Ram-Lak* filter.

In SAXSTT, it is a good approximation that all measurements of a particular point in reciprocal space lie on a great circle orthogonal to that point. Consequently, the real-space convolution of each point in reciprocal space will occur in the corresponding real-space plane orthogonal to that point. This means that we should in principle be able to apply a filter to the image formed by each detector segment, then compute the adjoint projection into 3D-reciprocal sphere space, and obtain a solution at least for the reciprocal space points sampled along a full semicircle. This reconstruction is likely to suffer heavily from missing wedges and similar artefacts for some points in reciprocal space, but it may be an interesting avenue to explore for very fast reconstructions that do not necessarily need to be very well-behaved. A comparable approach was demonstrated for diffraction grating dark-field tensor tomography by Kim *et al.* (2022) [38].

3.6.3 Relaxing the small-angle scattering constraint

It is interesting to consider what happens if we relax the small-angle scattering constraint, so that the transmitted and the scattered beam do not travel approximately the same path within the sample. In addition to the fact that we then probe a small circle, rather than a great circle, on the reciprocal space sphere, we will not be able to treat our projected data with transmission correction and directly regard it as the RSM. We rather have for our non-transmission-corrected data $d_h(s, j, k)$ that

$$d_h(s, j, k) = \int_{D'_h(s,\boldsymbol{l})\neq 0} \mathrm{d}\boldsymbol{l} D'_h(s, \boldsymbol{l}) \int_{-\infty}^{\infty} \mathrm{d}t\, \sigma(\boldsymbol{r}(s, j, k, t), \boldsymbol{p}(s), \boldsymbol{l}) f(\boldsymbol{r}(s, j, k, t), \boldsymbol{l}),$$

where σ is an opacity function, representing the absorption that the contribution to the measured scattering from each point in the field undergoes, depending on incident and scattering angles. Explicitly computing this function would be very expensive as it would be necessary to compute one value for every detector segment and projection direction for every point in the field. For example, for 300 projections and 32 detector segments, and a 50^3-size volume, it would be necessary to compute 1.2 billion parameters. However, observe that σ factors into an incident and an exiting component according to

$$\ln(\sigma(\boldsymbol{r}, s, \boldsymbol{l})) = \int_{-\infty}^{t} \mathrm{d}\tau a(\boldsymbol{r}_0 + \tau \boldsymbol{p}(s)) + \int_{t}^{\infty} \mathrm{d}\tau a(\boldsymbol{r} + t\boldsymbol{p}(s) + \tau \boldsymbol{p}'(s, \boldsymbol{l})),$$

where $\boldsymbol{p}'(s, \boldsymbol{l})$ is the scattering direction corresponding to the probed point on the RSM and the incident direction $\boldsymbol{p}(s)$. By considering the physical problem, we can conclude that the absorption undergone by an ensemble of photons that is exiting from the point $\boldsymbol{r}(s, j, k, t)$ is the same as that which a photon that travels *to* that point, going in the

opposite direction. This means that σ can be represented as the exponent of a field of *partial projection tensors* of the absorbance. The partially projected absorbance tensor can then be written

$$\boldsymbol{A}(\boldsymbol{r}, \boldsymbol{p}) = \int_{-\infty}^{t} \mathrm{d}\tau a(\boldsymbol{r}_0 + \tau\hat{\boldsymbol{p}}) \tag{3.20}$$

where $t = \left\|\boldsymbol{r} - \boldsymbol{r}_0\right\|_2$ and $\hat{\boldsymbol{p}}$ is an arbitrary projection direction vector. This leads to

$$\ln(\sigma(\boldsymbol{r}, \hat{\boldsymbol{p}}, \boldsymbol{l})) = \boldsymbol{A}(\boldsymbol{r}, \hat{\boldsymbol{p}}) + \boldsymbol{A}(\boldsymbol{r}, -\hat{\boldsymbol{p}}'(\hat{\boldsymbol{p}}, \boldsymbol{l})),$$

where σ is now a function of an arbitrary projection vector $\hat{\boldsymbol{p}}$, rather than one parameterized by s.

Unfortunately, evaluating Eq. (3.20) is significantly more difficult than carrying out the forward or adjoint John transform, since the discretized form of it combines aspects of both — a an accumulation of values that goes through real space but maps to each voxel. This means that it will be difficult to carry it out in slice-by-slice fashion due to the difficulty of interpolating values between different rays while maintaining independence between different blocks of the computation. One potentially successful approach would be to carry out a regular forward projection but storing the coordinates and values of each ray and at each step, and then carrying out a coordinate transformation after the fact to map the values to the coordinates of the voxels. Alternatively, one could simply compute one ray per voxel, as in the adjoint computation, and allow each of those rays to go through the partial forward projection necessary to reach the voxel. The first approach is likely more efficient since it does not require multiple computations for each path, but the second approach is likely more accurate since it does not involve an additional interpolation step.

Once the partial projections have been computed densely enough, the same approach to representing spherical functions that we have used for the RSM can be used to solve for the partial projection field in, *e.g.*, a Gaussian radial basis function representation. Given an easily pre-computed and evaluated expression for σ, the forward projection can then be computed similarly as the sparse John transform in Algorithm 6, binning the result directly into detector segments.

4

Implementation of tensor tomography in Mumott

Mumott is an open-source software package written in Python (available as the Python package `mumott` via PyPI) for analyzing tensor tomographic data. It is designed to be easy to install, portable, and have limited dependency on external libraries, while also providing high-performance implementations of tensor tomography algorithms. For numerics, Mumott depends primarily on NumPy, SciPy and Numba [39–41]. It also uses SciKit-Image for certain image-processing related features (such as cross-correlation alignment) [42], H5Py for input-output handling, and `tqdm` for progress tracking. Finally, it employs Matplotlib, as well as two libraries, ColorCET and `colorspacious`, providing colormaps suitable for scientific visualization [43]. In particular, the multiple-channel John transform (Sect. 3.3) is implemented using custom Numba kernels in both CPU and CUDA-based versions, and thus it has no dependency on external tomography libraries.

Mumott is designed with a wide range of possible users in mind, from experts working with high-performance computing setups to end-users working on laptops, and offers an extensively documented, modular, robust and thoroughly tested environment for exploring tensor tomography algorithms.

Fig. 4.1 shows the general outline of the object-oriented workflow in Mumott. This structure is characterized by object-oriented programming, while adopting some principles of functional programming. In effect, the parts of the implementation up to the ResidualCalculator function according to a typical inter-object communication structure, where objects prompt other objects to execute code by passing messages. The following parts including the interaction of the LossFunction, Regularizer, and Optimizer also have some things in common with functional programming. This is motivated by the optimization being a function of the loss function, which is the function of a residual and a set of regularizers. These two components operate independently

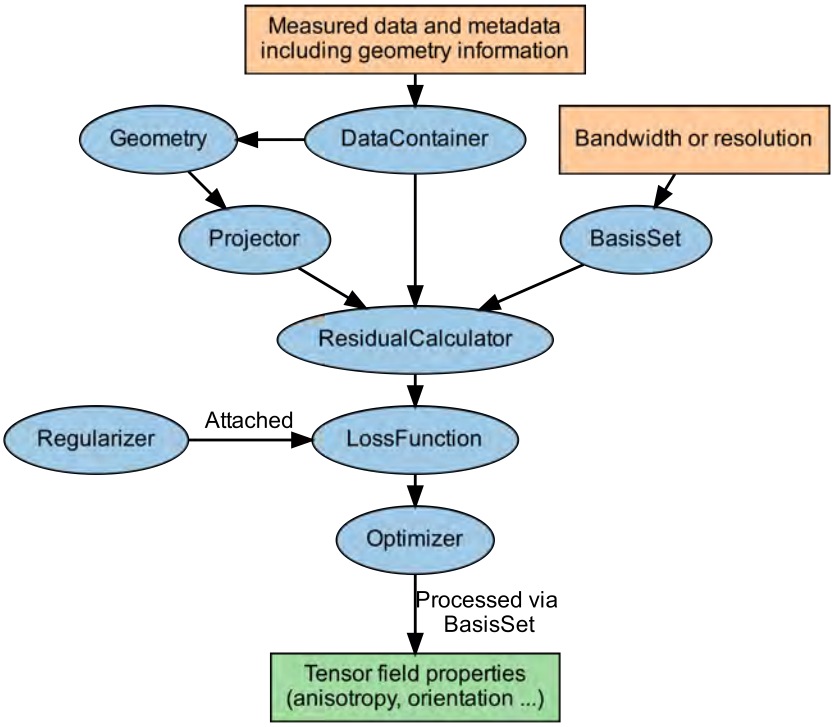

Figure 4.1: Outline of workflow of the object-oriented structure in MUMOTT. Orange boxes show input parameters and data, blue ovals show objects, the green box shows the output, and arrows indicate instances of objects interacting with one another.

of each other. As a result, the BASISSET, DATACONTAINER and PROJECTOR can be used independently of the optimization workflow.

Many objects are safely mutable after instantiation and employ hashes of their mutable properties to track the state of linked instances, which means that derived properties can be (automatically) recomputed when required. For the computation of hashes of floating-point numbers, each number is separated into a mantissa and an exponent. These are then rounded to five significant digits, before being concatenated and hashed using the Blake2B algorithm [44]. This allows hashes to be reproduced across different platforms with a high fidelity, even if the platforms do not yield exactly the same numerical result, and make changes between versions easy to identify through testing.

4.1 Input handling

Input data for MUMOTT is provided in the form of an HDF5 file, which contains a `projections` group, with numbered entries. Each numbered entry contains one "frame" of measured data. Additionally, it contains several basis vectors, rotation operators, and

detector segment angles.

4.1.1 Geometry

Along with measured projections, MUMOTT requires the specification of 5 basis vectors, as well as the central angle of each detector segment (which, as of version 2.0, are assumed to be of equal size). The 5 basis vectors, all in Cartesian coordinates, define the parameters of a projection measured when no rotation operation is being applied (*i.e.*, when both rotation and tilt angles are zero). These definitions assume that each projection is stored and accessed in contiguous, row-major order (also called C-contiguous), which is enforced by writing to and reading from the HDF5 format.

- Two coordinate systems are used in real space. The laboratory frame is spanned by three coordinates, (p, j, k). The laboratory basis vectors must be defined in terms of the Cartesian vectors (x, y, z), which are attached to the sample, and define directions*relative to a fixed sample.* Thus, the scanning directions are *the movement of the beam relative to the sample.* Moreover, *ordinality is defined in terms of contiguous row-major data ordering.* This means that the term *first* pixel index means *the index a change in which induces the largest jump in the image data* (because changing it skips to the next row), which in the HDF5 format means *the leftmost* index. This also means that what MUMOTT treats as the first scanning direction is *not* necessarily the fast scanning direction.[1]
  - $\hat{\boldsymbol{j}}$, the first scanning direction (when the first pixel index is incremented).
  - $\hat{\boldsymbol{k}}$, the second scanning direction (when the second pixel index is incremented).
  - $\hat{\boldsymbol{p}}$, the projection direction (the direction travelled by the beam impinging on the sample).
- Reciprocal space is spanned by three basis vectors, $\hat{\boldsymbol{q}} = (q_0, q_9 0, q_\parallel)$; see Sect. 3.2.2 for a more detailed discussion of these. At any given $\|q\|$, a circle in q-space is projected onto the detector,[2] which is spanned by two linear combinations of these three basis vectors. By definition, $\hat{\boldsymbol{q}}_\parallel = \pm\hat{\boldsymbol{p}}$, and so we define it as equal to $\hat{\boldsymbol{p}}$ by convention. This leaves the two other components, which are specified in terms

[1] At present, MUMOTT does not distinguish between different types of scanning. Although it would be possible to define the two directions in terms of the experimental fast and slow axis, it would ultimately only add to the burden of specification. Otherwise the user would need to not only track how the real-space geometry of their sample was set up, but also how it related to the experiment. In case this distinction is ever used, *e.g.*, for considering point spread functions, it would be easier for the user to simply specify whether j or k is the fast direction.

[2] Which may be a great or small circle on the sphere of constant $\|q\|$. In SAXS we take it to be a great circle per the small-angle approximation $\sin 2\theta \approx 0$.

of where the detector angle equals $\phi = 0^\circ$ and 90°, respectively. Two vectors must be specified in order to resolve whether ϕ rotates clockwise or anti-clockwise with respect to $\hat{\boldsymbol{p}}$. Like the other basis vectors they are specified in terms of the (x, y, z) basis vectors of the sample.

  - $\hat{\boldsymbol{q}}_0$, the direction of scattering at detector angle $\varphi = 0^\circ$.
  - $\hat{\boldsymbol{q}}_{90}$, the direction of scattering at detector angle $\varphi = 90^\circ$. The two *vêcq*-vectors together define the starting point of the detector coordinate system as well as whether the detector angle runs clockwise or counter-clockwise.

- There are typically two rotation axes in a SAXSTT experiment. It is not necessary to provide these in MUMOTT. Instead, a rotation matrix specifying rotation that they carry out can be included. However, MUMOTT supports their inclusion and enables some additional features (such as changing the axis of rotation or tilt for one or several measurements) if they are included. Moreover, they function as metadata, documenting a key aspect of how the experiment was carried out, and their inclusion is therefore encouraged.

  - The *outer axis*, also called $\hat{\boldsymbol{\beta}}$, is also called the tilt axis. It is fixed in the laboratory frame.
  - The *inner axis*, or $\hat{\boldsymbol{\alpha}}$, is also called the rotation axis or the standard tomographic axis. It moves as the rotation about the outer axis changes.

This coordinate specification system, is not minimalistic in the sense that it uses a larger number of vectors than is strictly necessary. This is because the system makes no assumptions about how a pixel index should correspond to a three-dimensional coordinate or how the angle of the detector should be defined. Instead, it is set up to allow a user or beamline scientist to refer to a photograph or schematic of the laboratory, set up whichever Cartesian coordinate system they are most comfortable with, check the ordering of their input data, and derive and specify the necessary MUMOTT vectors to carry out a reconstruction without any rearranging of their data.

An example for the structure of such an HDF5 file is shown in Table 4.1. The basis vectors and their corresponding field in the HDF5 file are also shown, along with the field names in the HDF5 format. Note that this structure is not exhaustive; for example, it is possible to supply an `inner_axis` and `outer_axis` for each projection, which will override any axes specified at the base level. It is also possible to specify a *rotation_matrix*, in which case the angle-axis pairs are not needed. If both are supplied, the provided rotation matrix is used unless the user decides to make changes to the geometry, which requires it to be re-calculated (by, *e.g.*, changing the associated angles).[3]

[3]It is the responsibility of the user to verify that the angle-axis pairs are consistent with the rotation matrix if data is provided in this redundant fashion. Given that rotation matrices often have relatively

Table 4.1: Top: Outline of the HDF5 file format used by MUMOTT. The spaces indicate the hierarchy of entries; `0` is an entry in the group `projections`, whereas `data` is an entry in the group `0`, and so on. Bottom: Unit vectors defining the experimental geometry of Fig. 4.2 and their values.

Path	Type
`p_direction_0`	`float`(3)
`j_direction_0`	`float`(3)
`k_direction_0`	`float`(3)
`detector_direction_origin`	`float`(3)
`detector_direction_positive_90`	`float`(3)
`inner_axis`	`float`(3)
`outer_axis`	`float`(3)
`volume_shape`	`int`(3)
`detector_angles`	`float`(n_φ)
`projections`	Group
`0`	Group
`data`	`float`(n_j, n_k, n_φ)
`diode`	`float`(n_j, n_k)
`inner_angle`	`float`(1)
`j_offset`	`float`(1)
`k_offset`	`float`(1)
`outer_angle`	`float`(1)
`weights`	`float`(n_j, n_k, n_φ)
`1`	Group
⋮	

Symbol	Value	Field name
$\hat{\boldsymbol{p}}$	$+\hat{\boldsymbol{z}}$	`p_direction_0`
$\hat{\boldsymbol{j}}$	$+\hat{\boldsymbol{y}}$	`j_direction_0`
$\hat{\boldsymbol{k}}$	$+\hat{\boldsymbol{x}}$	`k_direction_0`
$\hat{\boldsymbol{q}}_0$	$+\hat{\boldsymbol{x}}$	`detector_direction_origin`
$\hat{\boldsymbol{q}}_{90}$	$+\hat{\boldsymbol{y}}$	`detector_direction_positive_90`
$\hat{\boldsymbol{\alpha}}$	$+\hat{\boldsymbol{y}}$	`inner_axis`
$\hat{\boldsymbol{\beta}}$	$+\hat{\boldsymbol{x}}$	`outer_axis`

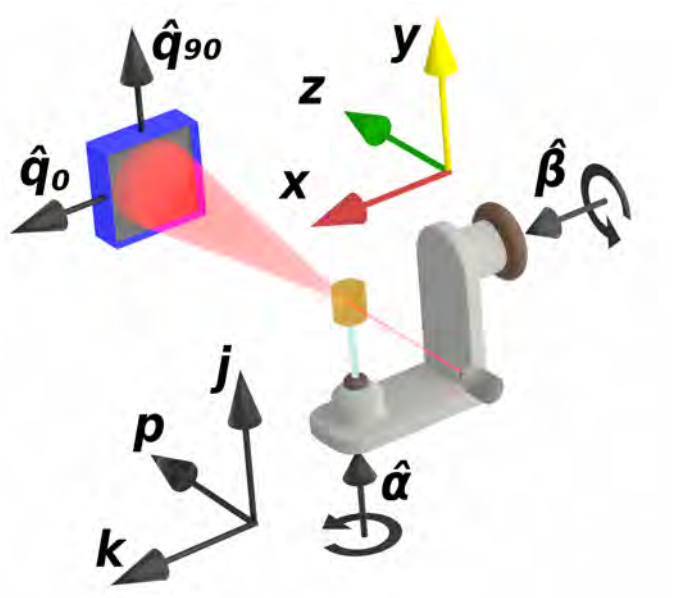

Figure 4.2: Examples of experimental setup and coordinate system. The coordinate system (x, y, z) is attached to the sample mounted on the rotation stage, whereas the system (p, j, k) is attached to the laboratory frame. The vector pair $(\hat{\boldsymbol{q}}_0, \hat{\boldsymbol{q}}_{90})$, which defines probed directions in reciprocal space, is also attached to the laboratory frame. The rotation axis $\hat{\boldsymbol{\beta}}$ is fixed to the laboratory frame, while $\hat{\boldsymbol{\alpha}}$ rotates as the sample is rotated about $\hat{\boldsymbol{\beta}}$. Figure adapted from Paper IV.

The differently numbered projections do not need to have the same shape in the (n_j, n_k) dimensions; the input will be padded to the largest size. The reconstruction will then be performed within the user's preferred coordinate system.

An example of such a system is shown in Fig. 4.2. This leaves the definition of the detector angle as the main potential uncertainty for the user. It can be derived given adequate knowledge of the technical specifications of the detector and the data reduction pipeline. However, the detector segment orientations should ideally be checked by using a test sample with known scattering directions, such as a carbon fiber knot. For each projection, then, the rotation operation to be applied to the basis vectors needs to be provided, as a rotation matrix or as two axis-angle pairs. Finally, the geometry requires offset factors in the **j** and **k** directions to be either provided or determined via an alignment algorithm. These offset factors should eliminate precession, ensuring that all projections share a common center of rotation.

low numerical accuracy, and that axis-angle pairs are useful as metadata, we opted to neither carry out a check of nor to forbid this usage. Instead, the log will tell the user whether the internally used rotation operators were taken from provided matrices or generated from axis-angle pairs.

4.1.2 Data

As shown in Table 4.1, a single frame of projection data in MUMOTT is a 3-dimensional array with the dimensions (n_j, n_k, n_φ) — number of pixels in the first direction, number of pixels in the second direction, and number of detector segments per pixel. In addition, projection data needs to be transmission corrected and therefore a diode entry with dimensions (n_j, n_k) is also expected. If the projection data is already transmission corrected, it is not strictly necessary to provide the diode data, however, it is useful for aligning projection frames. The user may also provide a set of weights of the same shape as the projection data, which can be used to mask data points known to be invalid as well as to encode uncertainties.

4.2 Method objects

The forward model in MUMOTT is specified by composing method objects. Specifically, these objects include real-space linear mapping objects (PROJECTOR), reciprocal-space linear mapping objects (BASISSET) as well as the RESIDUALCALCULATOR object.

4.2.1 Projectors

The PROJECTOR objects do not, in the strict mathematical sense, carry out a projection, which is typically defined as an idempotent mapping onto a subset. Rather, they compute the John transform of the tensor fields (Eq. (2)) and its adjoint (Eq. (4)), which can be regarded as being composed of a *series* of projections.

A projector object is initialized by giving it a GEOMETRY instance. The necessary code for computing the John transform and its adjoint is then compiled, and can be executed on request through the methods `.forward(field)` and `.adjoint(image)`. There are two types of projectors: a SAXSPROJECTOR and a SAXSPROJECTORCUDA. The SAXSPROJECTOR uses CPU-based Numba kernels to compute the John transform while a SAXSPROJECTORCUDA instance uses the Numba CUDA interface to compile and execute kernels written for the GPU, which can be several orders of magnitude faster.

4.2.2 Basis sets

A BASISSET is an object for computing and applying a mapping between a representation on a sphere to a representation on a detector. This is essentially the procedure described in Sect. 3.2.3. Currently there are three BASISSET implementations in MUMOTT . The BASISSET functions analogously to the PROJECTOR, with a `forward` and a `gradient` method, except that it applies to the reciprocal space part of the representation.

4.2.3 Residual calculators

In order to compose the forward model, the ResidualCalculator is used. It takes a BasisSet, a Projector as well as a DataContainer as inputs. Its main function is to compute the residual, which we may symbolically write as

$$\boldsymbol{r} = \mathbf{B}\left(\mathbf{P}\left(\boldsymbol{X}\right)\right) - \boldsymbol{D},$$

where $\mathbf{B}$ represents the forward action of the BasisSet, $\mathbf{P}$ represents the forward action of the Projector, $\boldsymbol{X}$ represents the reconstruction, and $\boldsymbol{D}$ is the data.

The ResidualCalculator also computes the gradient of the residual norm with respect to $\boldsymbol{X}$, *i.e.*,

$$\nabla_{\boldsymbol{X}} \left\|\boldsymbol{r}\right\| = \mathbf{B}^T\left(\mathbf{P}^T\left(\nabla_{\boldsymbol{r}} \left\|\boldsymbol{r}\right\|\right)\right),$$

where $\mathbf{B}^T$ represents the adjoint operation (or more generally, gradient) of $\mathbf{B}$ and $\mathbf{P}^T$ represents the adjoint of the John transform.

In addition to providing these operations, the ResidualCalculator tracks and dynamically updates the current state of $\boldsymbol{X}$.

4.3 Optimization objects

In order to obtain a solution to a system, we need to define an optimization. The optimization approaches supported by this framework are gradient-based methods of the form

$$\mathcal{L}(\boldsymbol{X}) = \left\|\mathbf{B}(\mathbf{P}(\boldsymbol{X})) - \boldsymbol{D}\right\|_a^a + \lambda \left\|F(\boldsymbol{X})\right\|_b^b + \ldots$$

where the ellipsis indicates that more terms of the form $\lambda \left\|F(\boldsymbol{X})\right\|_b^b$ may be added, where λ is a regularization weight, $F(\dot{})$ is some function of $\boldsymbol{X}$, and a and b specify norms.

4.3.1 Loss functions

The loss function is defined through a LossFunction object, which takes the ResidualCalculator as input. There are currently two LossFunctions. The SquaredLoss is used for ordinary least-squares regression, whereas the HuberLoss can be used for robust regression. Instances of these classes expose a get_loss() method, which can be used to obtain the sum of the residual norm and the regularization terms as a function of a reconstruction $\boldsymbol{X}$.

In addition, the LossFunction can be given Regularizers along with the respective regularization weights. It will then apply the regularizing contributions to the loss and its gradient when get_loss is invoked.

4.3.2 Regularizers

A number of Regularizers are implemented in Mumott. These interact with the LossFunction by providing it with a regularization norm and a regularization gradient, which are used together with the regularization weight to find the solution to the optimization problem. These include:

Laplacian – Smooths the solution by minimizing the squared L_2 norm of the Laplacian of the tensor field.

TotalVariation – Smooths each coefficient of the solution in a more robust manner than the Laplacian, by minimizing the Euclidean norm of the gradient for each coefficient. Can be configured to use the Huber approximation for small values, in order to make convergence easier.

L2Norm, L1Norm – Minimizes the L_2 and L_1 norms of the solution, respectively. The L_2 norm penalizes large values (ridge regression), while the L_1 norm penalizes small values, yielding a sparse solution (LASSO).

HuberNorm – Minimizes the Huber norm of the tensor field, acting as an L1Norm for large values and an L2Norm for small values.

4.3.3 Optimizers

There are two general-purpose optimizers in Mumott, along with a special zonal harmonics optimizer.

GradientDescent – Uses fixed-stepsize gradient descent, with an option to use Nesterov accelerated momentum.

LBFGS – Uses the SciPy implementation of the LBFGS-B algorithm for quasi-Newton solution of the optimization.

ZonalHarmonicsOptimizer – A special optimizer for reconstructions using zonal harmonic representations of the RSM.

4.4 Pipelines

There are reasons for why some users might not necessarily want to employ the object-oriented interface, at least not initially. One reason typically occurs to novice users – the object-oriented interface is complicated, and therefore one desires a simplified interface. Another reason that is more likely to occur to experienced users is that some features are not practical or feasible to implement within the object-oriented interface, as it

presently exists. One such case is alignment, where the data set needs to be aligned such that the lines of integration in the John transform map correctly to three-dimensional coordinates.

4.4.1 Alignment

Phase matching alignment – Uses a simple cross-correlation algorithm which compares the data with forward projections, to estimate the offsets to the coordinates (j, k) which are needed to align projections to each other. Can efficiently upsample projections by using a refined cross-correlation calculation [45].

Optical flow alignment – The optical flow alignment uses several methods to improve and refine the phase matching alignment approach [46].

4.4.2 Standard reconstruction

MITRA – Modular iterative tomographic reconstruction algorithm (MITRA) is a highly modular pipeline that by default uses Nesterov accelerated gradient descent, a GAUSSIANKERNELS representation as well as "tensor SIRT" weights and preconditioners. However, it can take a wide range of keyword arguments, allowing for it to be customized. It is therefore a suitable interface for the intermediate user to experiment with.

SIGTT – The spherical integral geometric tensor tomography (SIGTT) pipeline combines a basic SPHERICALHARMONICS basis set and a LAPLACIAN regularizer.

Discrete directions – This pipeline is used to emulate the approach of Schaff *et al.* [20], by splitting up the reconstruction into multiple sub-datasets.

4.4.3 Asynchronous reconstruction

There are several pipelines which execute asynchronously on the GPU, avoiding synchronization overhead from transferring data between the CPU and GPU. These include a tensor SIRT pipeline, which is similar to MITRA without Nesterov momentum. In addition, there is MOTR, which is essentially the default MITRA pipeline with two fixed regularizers attached, for L_1 and two-sided total variation regularization. robust and denoised tensor tomography (RADTT) optimizes for the Huber norm with two-sided total variation regularization, making it more robust to noise. There are also sparse versions of the asynchronous pipelines, which use VRAM more effectively. For details on asynchronous and sparse approaches to reconstruction, see Sect. 3.4.

Table 4.2: Comparison of reconstruction times in seconds averaged across 10 runs each for a typical single-q dataset consisting of 247 projections, each with 65×55 pixels and 8 detector segments, using different reconstruction pipelines and running on different computers. N is the number of basis functions per voxel. In all cases, relative uncertainties were smaller than 5%, and are omitted to maintain ease of reading. The workstation (WS) data was obtained using an AMD Ryzen 7 3700X processor with 8 physical cores, 64 GB of DDR4 2666 MHz RAM, and for the GPU accelerated calculations an Nvidia GeForce RTX 3060 GPU with 12 GB of VRAM. The high-performance computing (HPC) CPU timings were generated using 8 top-level threads on a 64-core Intel Xeon Platinum 8358 @ 2.0 GHz CPU with some operations utilizing lower-level multithreading. The HPC GPU timings were obtained used an Nvidia A100 GPU with 40 GB of VRAM and 8 threads on 16 cores of a 64-core Intel Xeon Platinum 8358 @ 2.0 GHz.

		CPU		GPU	
	N	WS	HPC	WS	HPC
SIGTT	6	23	18	9	9
	20	45	29	18	14
	72	108	69	60	36
MITRA	18	41	22	13	8
	50	93	45	40	14
	162	271	156	115	37
DD	18	81	72	40	43
	50	156	157	101	111
	162	334	392	290	346
MOTR	18			9	8
	50			12	10
	162			46	22

4.5 Timing and evaluation

It is instructive to look at the performance of various pipelines for different platform; see Table 4.2 for a comparison between a CPU and GPU-using, typical mid-range workstation, and a high-end high-performance computing (HPC) node. GPU-based computations are very fast, especially for the HPC node compared to the workstation. We can also look at the difference in computational time for different SAXSTT methods, in particular the computing times for the approaches of Liebi *et al.* (2015, 2018) and Gao *et al.* (2019) and Mumott 0.2, as reported in Paper I, to methods in the current version of Mumott [21, 22, 47, 48].

Table 4.3: Comparison of reconstruction times for different SAXSTT methods. N is the number of coefficients per voxel that are being optimized. The speedup is computed relative to the speed of the zonal harmonics method of Liebi *et al.* (2015, 2018). The methods of type CPU are computed using only CPU. The methods of type CPU/GPU use the GPU for the John transform computations, but CPU resources for everything else. MOTR, the method of type GPU, uses only GPU resources and executes asynchronously. ZH, IR, and SIGTT (Mumott 0.2) were run on the workstation described in Paper I, with a 12-core AMD Ryzen 3900x CPU. The remaining entries were run on the same workstation as described in Table 4.2.

	N	Time (min)	Speedup	Type
ZH (Liebi *et al.* 2015)	6	500	1	CPU
IR (Gao *et al.* 2019)	6	26.7	18	CPU
SIGTT (Mumott 0.2)	28	6.7	75	CPU
SIGTT (Mumott 2.0)	28	2.05	243	CPU/GPU
MITRA (Mumott 2.0)	72	1.97	254	CPU/GPU
MOTR (Mumott 2.0)	72	0.38	1304	GPU

This comparison is shown in Table 4.3. The speedups and timings are only intended as a rough guide to the improvement in performance for SAXSTT throughout the development of the methods. Because the different methods have different approaches to optimization, termination and regularization as well as considerable differences in their final result and the number of coefficients which are being optimized, it is difficult to carry out a one-to-one comparisons of their performance. Some judgment is therefore necessary in determining what parameters are "reasonable" for each reconstruction. The timing differences are, however, several orders of magnitude, demonstrating that Mumott provides a significant improvement in performance.

5

Validation of SAXSTT

Many assumptions are required for SAXSTT, including ones related to beam quality, the applicability of reciprocal space mapping to a local area, smoothness in real and reciprocal space, the first Born approximation holding, and so on. Not all of these assumptions are easy to check, and it is even more difficult to verify that all assumptions taken together hold. Thus, a set of validation techniques must be employed. Within the context of this work, papers I and II are concerned with validation.

First, there are basic "sanity checks", that is, checking that the problem is posed in a reasonable way and that the reconstruction can recreate the measured data. Sanity checking includes looking at slices of the reconstruction, and comparing synthetic data computed from the reconstructions to the actual data (which may be real or simulated). Sanity checks are a routine part of reconstruction, and examples can be seen in, *e.g.*, Liebi *et al.* (2015) [21]. In this category we may also include *robustness* checks, that is, seeing that the reconstruction behaves reasonably under perturbation and checking whether it consistently converges toward a particular solution. Robustness is one of the topics of Paper I, where we investigated reconstruction under added noise, and looked at the variation within an ensemble of reconstructions.

Second, there is validation through simulation, where a simulated sample is created and data is generated from it. The degree of accuracy to experiment can be variable - generally, one would start out with a relatively simple simulation. Simulation validation is important, because it can be used to explore the limits of the method under ideal, or strongly controlled conditions. Moreover, it can be used to very precisely compute the degree to which the reconstruction represents the simulated sample. Validation through simulation is the principal topic of Paper I, as it investigates the accuracy of several SAXSTT methods applied to simulated data.

Finally, the is experimental validation, which is necessary to account for all the assumptions made with respect to experimental parameters. This category essentially

includes all work where previous knowledge exists about the sample, such as the work of Liebi *et al.* (2015) and Schaff *et al.* (2015) [20, 22], where one can see *e.g.* the trabecular bone orientations line up with the gross structure of the sample. In particular, however, we mean by experimental validation work such as that of of Guizar-Sicairos *et al.* (2020) [49], which follows up on Liebi *et al.* (2015) and compares a SAXSTT reconstruction of a whole sample to SAXSTT reconstructions of individual slices of the same sample. Experimental validation is the topic of Paper II, where we look at the effects of the missing wedge problem under typical experimental conditions, compared to a complete data set.

5.1 Robustness

One of the basic ways that the reliability of a reconstruction can be checked is to carry out the reconstruction multiple times, with small perturbations to the initial conditions, and check whether the result is consistent across the resulting *ensemble* of reconstructions. In this case, we are comparing the SIGTT method as implemented in Mumott 0.2 with the zonal harmonics (ZH) method of Liebi *et al.* (2015, 2018) [21, 22] and the iterative reconstruction (IR) method of Gao *et al.* (2019) [47], using the trabecular bone sample labelled B in Liebi *et al.* (2015). We carry out 10 reconstructions, randomizing the angles which are required for ZH, and adding small amounts of noise to the initial condition of SIGTT and IR, since these methods do not depend on angle parameters. The 10 reconstructions are compared by computing the quotient

$$Q(\boldsymbol{r}) = \frac{\operatorname{var}\left(\frac{1}{10}\sum_{n=1}^{10}\boldsymbol{a}_n(\boldsymbol{r})\right)}{\frac{1}{10}\sum_{n=1}^{10}\operatorname{var}\left(\boldsymbol{a}_n(\boldsymbol{r})\right)} \tag{5.1}$$

where $\boldsymbol{a}_n(\boldsymbol{r})$ is the spherical function representation in one voxel in a reconstruction within each ensemble. The variance over the sphere, $\operatorname{var}\left(\boldsymbol{a}_n(\boldsymbol{r})\right)$ is a measure of the power of the anisotropic component of the reciprocal space map, and it is calculated using the spherical harmonic cross-spectral theorem; see subsubsection 3.2.1.2 and in particular Eq. (3.7). If the reconstructions are all identical, then $Q(\boldsymbol{r})$ will equal 1 everywhere. If the reconstructions are different, then it will be smaller than 1, as some of the anisotropic components will cancel out when the different iterations are summed up before the variance is calculated. Orientations and relative anisotropies are computed from the averaged reconstructions. The orientation determination for ZH and SIGTT, and the conversion of the rank-2 tensor of IR to spherical harmonic form uses the isomorphism to $\ell_{\max} = 2$ spherical harmonics; see Eq. (3.9). The relative anisotropy is

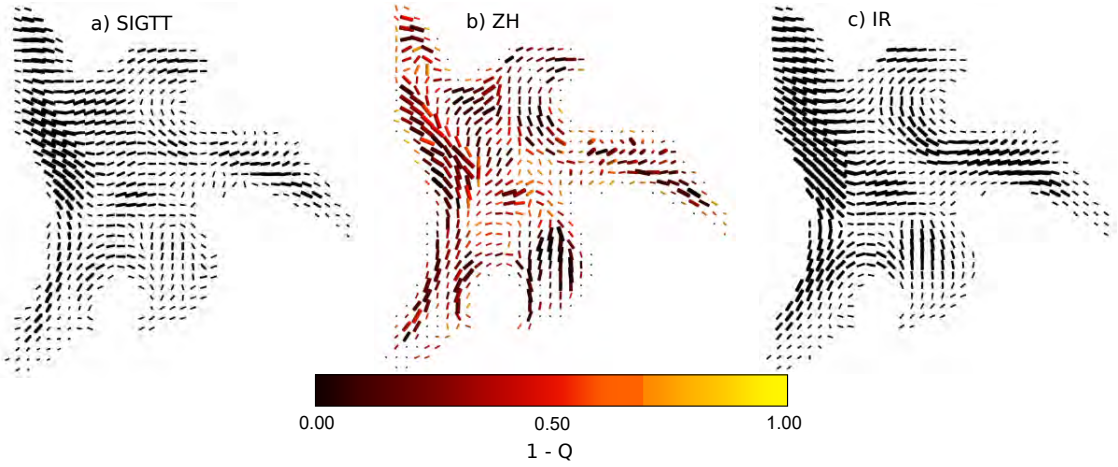

Figure 5.1: Experimental ensemble reconstructions. a) Virtual slice from ensemble of 10 reconstructions each with randomized initial conditions of a sample of trabecular bone using SIGTT. **b)** Ensemble reconstruction using ZH. **c)** Ensemble reconstruction using IR. The glyphs are scaled by the square root of the spherical variance, defined in Eq. (5.2). The quantity Q, defined in Eq. (5.1), is a measure of how much the anisotropy of each RSM changes across the ensemble of reconstructions. The methods generally agree in terms of the orientation of each RSM, but only ZH shows a change in the anisotropy across the ensemble.

the standard deviation over the sphere normalized by the mean,

$$\zeta(\boldsymbol{r}) = \frac{\sqrt{\operatorname{var}(\boldsymbol{a}(\boldsymbol{r}))}}{\overline{\mu}(\boldsymbol{a}(\boldsymbol{r}))}. \tag{5.2}$$

where $\overline{\mu}(\boldsymbol{a}(\boldsymbol{r}))$ is the spherical mean, given by the spherical harmonic coefficient a_0^0.

The results of this comparison are seen in Fig. 5.1. The figure shows that both SIGTT and IR yield consistent results over the entire ensemble. However, ZH does not. This is not surprising – the zonal harmonics do not span a linear subspace of the spherical harmonics. This means that the sum of two expansions in zonal harmonics is not necessarily itself zonal (only if the two zonal expansions have the same main axis). For this reason, the ZH reconstruction is liable to get stuck in local minima.

5.2 Simulations

In order to make the simulated samples shown in Fig. 5.2 mimic real samples to a reasonable extent, several elements needed to come together. First, the simulated samples needed to be structured in reasonable ways – for example, they needed to have some amount of continuity in real and reciprocal space, so some kind of correlation needed to be enforced. Second, in order to permit the use of non-convex samples, the

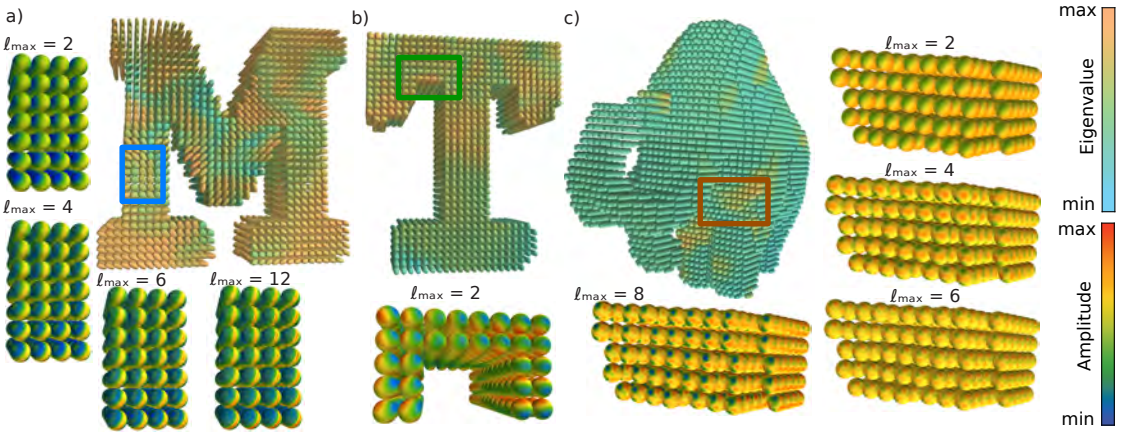

Figure 5.2: Superquadric glyph render of simulations. **a)** Sample "M" with its zonally symmetric RSMs from one region (blue square), truncated at $\ell = 2,\ 4,\ 6,\ 12$ respectively. **b)** Sample "T" with its rank-2 tensor RSMs from one region (green square). **c)** Sample "mammoth" with its unrestricted $\ell_{\max} = 8$ RSMs from one region (orange square), truncated at $\ell = 2,\ 4,\ 6,\ 8$, respectively. The colors of the superquadric glyph renders show the largest eigenvalue of the rank-2 tensor component of each simulated sample, and the upper and lower bounds of the color mapping are individually set to reveal the texture of each sample. The colors of the RSM renders show their amplitudes and are scaled to the maximum and minimum amplitude in the selected region of each sample.

metric used to determine the continuity should be an interior metric – if the metric used was simply, *e.g.*, the euclidean norm, then it would not be possible to properly model discontinuities in real space. Third, the real-space distribution of amplitudes should have some kind of larger-scale structure, and not be too "random". With these concerns in mind, the following model was devised:

- A real-space mask of valid voxels was constructed from extruded letters in two cases, and a three-dimensional render of a mammoth in one case.
- Using a variation on Dijkstra's algorithm with a step size of 2.5 [50], combined with a k-d tree to ensure that all points are checked in order of distance and each point is checked only once [51], the approximate interior distance from every source point to every other point was determined.
- A number of source points was chosen for each sample, and distributed at large distances. Specifically, four sources for "M", two source points for "T", and five source points for "mammoth".
- A real-space distribution was assigned to each source point, with a Gaussian decay, large enough that the different sources had substantial overlap.

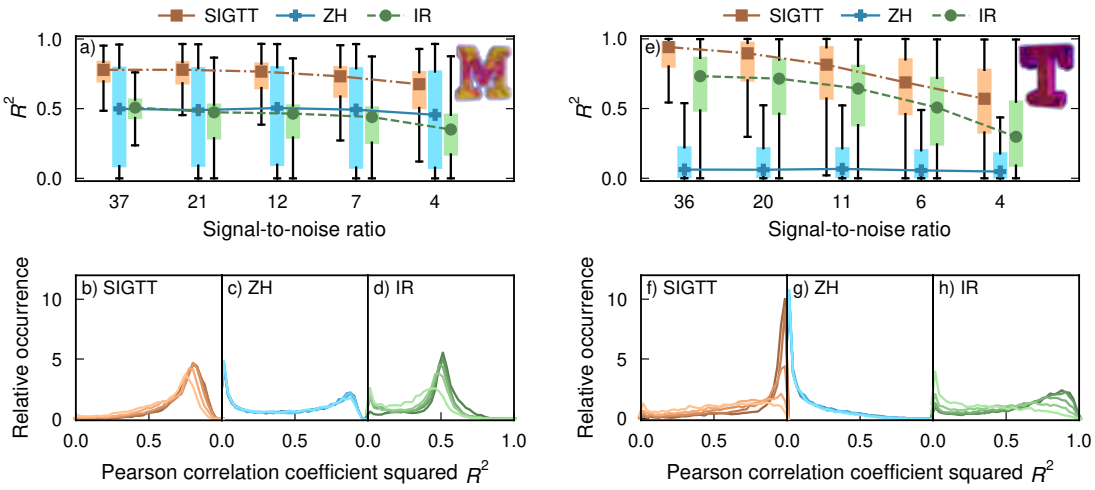

Figure 5.3: Correlations for sample "M" and "T". a) Box plots of R^2 as defined in Eq. (3.8), with lines and symbols indicating the respective median of each box plot. Outlier dots each represent 100 voxels. **b)–d)** Correlation coefficient distribution for each of the three methods. **e)** Box plots of R^2. **f)–h)** Correlation coefficient distribution for each of the three methods. The signal-to-noise ratio (SNR) goes from 37 ("M") or 36 ("T"), shown with the darkest lines, down to to 4, shown with the lightest lines. The image inset shows a volume render of each simulated sample.

- A model RSM was assigned to each source point. The RSM all had specific limits on their band-limits - $\ell_{\max} = 12, 2, 8$ for "M", "T", and "mammoth", respectively. In the case of "M", the source points were all zonal, following a decaying power law to ensure a single great-circle band of intensity.

- An optimization was then carried out with respect to a loss function combining nearest-neighbour correlation, distance-weighted correlation to source points, distance weighted-averaged similarity to source points in terms of the spectral power, and in the case of "M", enforcement of zonal symmetry.

The zonal symmetry in the case of "M" was enforced by requiring correlation with the ℓ-weighted Dirac delta function,

$$w(\ell)\delta_m^\ell(\theta,\phi) = (-1)^{\ell/2} Y_\ell^m(\theta,\phi),$$

separately for each frequency band ℓ. The purpose of "M" and "T" was to follow the types of symmetries assumed by the ZH and IR approaches respectively, whereas "mammoth" represented a more general symmetry. Following this, the three samples were reconstructed using each of the three methods.

Figure 5.4: Comparison of virtual slices of "M". a) Virtual slice of simulated sample "M". **b)** SIGTT reconstruction. **c)** ZH reconstruction. **d)** IR reconstruction. The glyphs are colored according to the relative anisotropy, Eq. (5.2), and scaled according to the spherical mean of each RSM. All three reconstructions follow the orientations of the model reasonably well on average. The SIGTT reconstruction follows both the orientations and relative anisotropy of the model closely. The ZH reconstruction has a lot of variation in the relative anisotropy, as well as many orientations deviating from the overall tendency to follow the model. The IR reconstruction follows the model almost as well as the SIGTT reconstruction.

The results of the reconstruction of "M" and "T" are showin in Fig. 5.3. In all cases, SIGTT is the best performing method. In the case of "T", ZH is clearly the best performing method, but the case of "M" is more ambiguous, since ZH has a slightly higher median for the lower SNR cases, but also a larger dispersion. IR is bounded above at a correlation of around $R^2 = 0.5$, since for sample "M", the $\ell = 2$ component, to which the rank-2 tensor of IR is isomorphic, contributes only about half of the variance of each RSM. Although SIGTT performs better than IR for sample "T", the representations used by each method is in fact equivalent. The improvement in the reconstruction quality for SIGTT is most likely due to its use of regularization.

In Fig. 5.4, glyph renders for each reconstruction of "M" are shown. SIGTT and IR both follow the model closely, although the relative anisotropy of IR is low in some places. ZH reconstructs the orientations of the model reasonably well, but performs less well in terms of reconstructing the relative anisotropy.

We can finally see the quality of the reconstruction of "mammoth" in Fig. 5.5. In this case, SIGTT performs by far the best. This is not surprising, since "mammoth" lacks the zonal symmetry assumed by ZH, and only has a weak $\ell = 2$ component for IR to reconstruct.

5.3 The missing wedge problem

The so-called missing-wedge or limited-angle problem is an issue that occurs when tomographic measurements are absent or not sufficiently dense along a particular trajectory, which can be derived from the projection-slice theorem of tomographic reconstruction [52, 53]. In order to investigate this effect in SAXSTT, we carried out

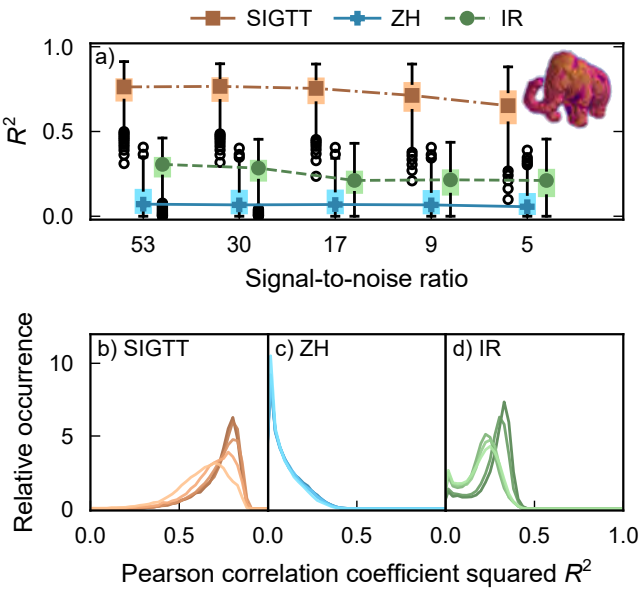

Figure 5.5: Correlations for sample "mammoth". a) Box plots of R^2 as defined in Eq. (3.8), with lines and symbols indicating the respective median of each box plot. Outlier dots each represent 100 voxels. **b)–d)** Correlation coefficient distribution for each of the three methods. The SNR goes from 53 (darkest lines) down to to 5 (lightest lines). The image inset shows a volume render of the simulated sample.

an experiment with a novel acquisition scheme, designed to quantify the effect of missing wedges in a typical acquisition. We chose human trabecular bone as the model system, since it has regular symmetries and is continuous in the RSM, and is known to reconstruct well even when the data has been reduced into only 8 segments (see Liebi *et al.* (2015, 2018) and Paper I of this work [21, 22]). For the reconstruction, we used the GaussianKernels representation and the pipeline MITRA, together with a Huber norm and coefficent-wise total variation regularization.

Per the discussion in Sect. 2.2, we can expect to reconstruct components within a 45° cap around the main tomographic axis. The quality of the remaining reconstruction can be quantified by computing a quality factor – if we assume that the densely sampled parts of the sample are well-sampled, we can define a sample density as

$$\rho(\hat{\boldsymbol{p}}') = \begin{cases} 1 & \text{if the direction } \hat{\boldsymbol{p}} \text{ of the nearest} \\ & \text{measurement satisfies } \Delta(\hat{\boldsymbol{p}}', \hat{\boldsymbol{p}}) < \delta, \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}$$

where $\Delta(\boldsymbol{v}, \boldsymbol{u})$ is the Friedel symmetric great-circle distance, defined as

$$\Delta(\boldsymbol{v}, \boldsymbol{u}) = \arccos\left(\frac{|\boldsymbol{v} \cdot \boldsymbol{u}|}{\|\boldsymbol{v}\| \, \|\boldsymbol{u}\|}\right), \tag{5.4}$$

The limitation in measurement can be overcome by introducing a third rotation axis, about the direction of projection, $\hat{\boldsymbol{p}}$. This was done by taking the sample off the holder, turning it by 90° about $\hat{\boldsymbol{p}}$, and putting it back on a new needle. The quality factor is computed from this density by subjecting it to the Funk-Radon transform, which has the general definition

$$F[g](\hat{\boldsymbol{v}}) = \int_L \mathrm{d}\hat{\boldsymbol{l}} g(\hat{\boldsymbol{l}}) \quad \text{with} \quad \left\{\hat{\boldsymbol{l}} \in L \mid \hat{\boldsymbol{l}} \perp \hat{\boldsymbol{v}}\right\} \tag{5.5}$$

In other words, the Funk-Radon transform consists of taking the line integral over all great circles, and mapping the result of each of those integrals to the point on the sphere orthogonal to that great circle [29]. Thus we obtain the missing wedge quality factor

$$\kappa(\hat{\boldsymbol{v}}) = F[\rho](\hat{\boldsymbol{v}}). \tag{5.6}$$

Because SAXS probes points on the RSM perpendicular to the direction of the impinging beam, the Funk-Radon transform of ρ in Eq. 5.3 will indicate how complete the tomographic data set of each point in the RSM is. We thus define this quantity $F[\rho]$ as our RSM quality factor.

Fig. 5.6a)–c) shows the distribution of individual and combined measurements - data set 1 before the rotation and remounting, data set 2 after the remounting, and the full, combined data set. The panels d)–f) show the quality factor, indicating that for data set 1, the quality is lower near the equator $q_y = 0$, whereas for data set 2, the quality is lower along the meridian $q_x = 0$. For the full data set, there is no loss in quality, since the measurements have been carried out over the entire sphere.

In Fig. 5.7 we see the distribution of errors in reciprocal space, both in terms of the coefficients of the basis functions, and in terms of the amplitude of the RSM. Pearson's $\boldsymbol{R}^2$ is, as before, given by Eq. 3.8, which is valid in general, and not just for a spherical harmonic representation. The basis functions are overlaid the quality factor from Fig. 5.6. By necessity, the error for the basis functions is higher due to their non-orthogonality; some of the errors cancel out when each basis function is projected onto the sphere.

We can see the full reconstruction of the sample in Fig. 5.8, using a spherical function glyph render with the shape given by the Funk-Radon transform of the amplitude. Viewing the three reconstructions from a distance, we can observe that they are very similar, thus suggesting that the missing wedge problem has a limited overall impact on the reconstruction. However, viewed close up, the impact of the problem becomes more apparent, as certain spherical functions are distorted relative to the full data set.

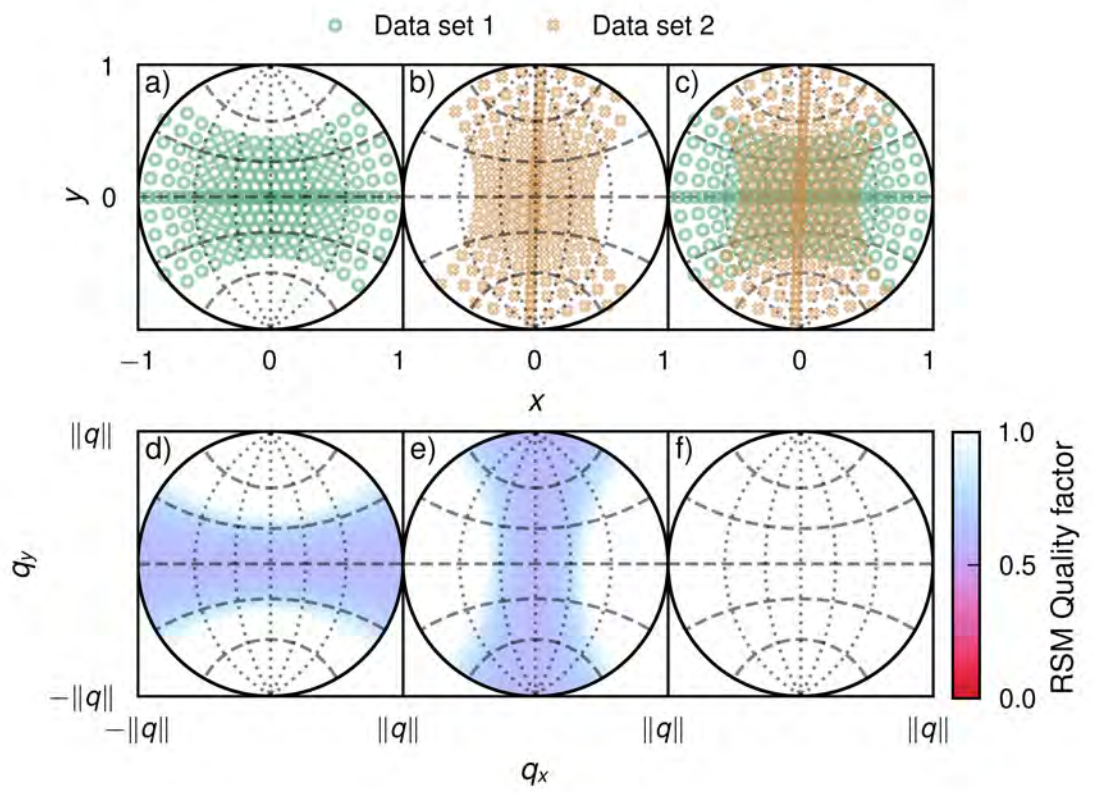

Figure 5.6: Points on hemisphere of projection and theoretical quality factors. a) Probed points on sphere of projection in first measurement. **b)** Probed points in second measurement. **c)** Combined points from both measurements. **d)** Quality factor in reciprocal space from first data set. **e)** Quality factor from second data set. **f)** Quality factor from combined data sets.

In particular, smearing and amplitude reduction (which is also reflected in a reduction in the elongation of the shapes) along the directions where the quality factor is lower is observed.

In Fig. 5.9, we see a comparison of orientation errors for data set 1 and 2. The orientations are quite robust in general, especially for places where the relative anisotropy is hig, such as for the straight, pillar-like structures. There are more errors in flatter and interface-like areas, where the relative anisotropy also tends to be lower. The zoom-in shows an interface area where the orientation error is relatively high; at the center, the orientations appear almost scrambled. The more horizontal (x-aligned) orientations appear better determined in data set 1, while the more vertical (y-aligned) orientations appear better determined in data set 2. This is consistent with the y-component of the RSM being better determined in data set 1, and the x-component being better determined in data set 2.

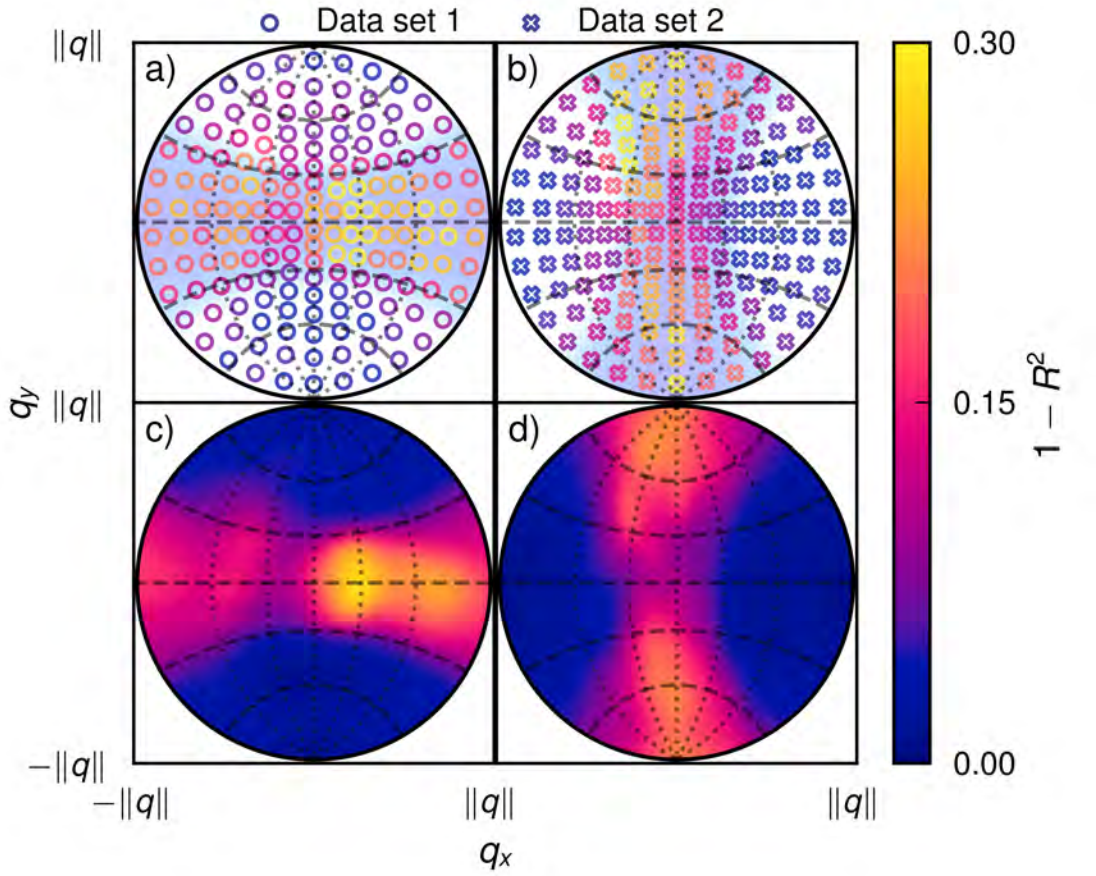

Figure 5.7: Error distribution for each RSM basis function. a) Coefficient errors for data set 1. **b)** Coefficient errors for data set 2. **c)** RSM errors for data set 1. **d)** RSM errors for data set 2. The markers have been placed over the corresponding theoretical quality factor from Fig. 5.6. The errors in **a)** and **b)** were calculated by computing the overall Pearson correlation coefficient for each basis function coefficient when comparing the partial and full data sets. In **c)** and **d)** the correlation coefficients were computed for the RSM amplitude at each coordinate.

We can conclude from this investigation that the missing wedge problem is present in SAXSTT, and that its effect on reconstructions can be inferred from tomographic theory, by considering the density of measurements in each area. However, we note that the impact of the missing wedge problem is limited on derived properties, *e.g.*, the main orientation of anisotropic RSMs (as can be seen from the glyph scaling in Fig. 5.9). Moreover, its effect can apparently be limited by considering the orientation of the nanostructure during data acquisition, and accounting for the when mounting the sample, such that part of the amplitude of the RSM ends up in a high-quality region. Finally, it is plausible that the imposition of stronger symmetry constraints on the RSM,

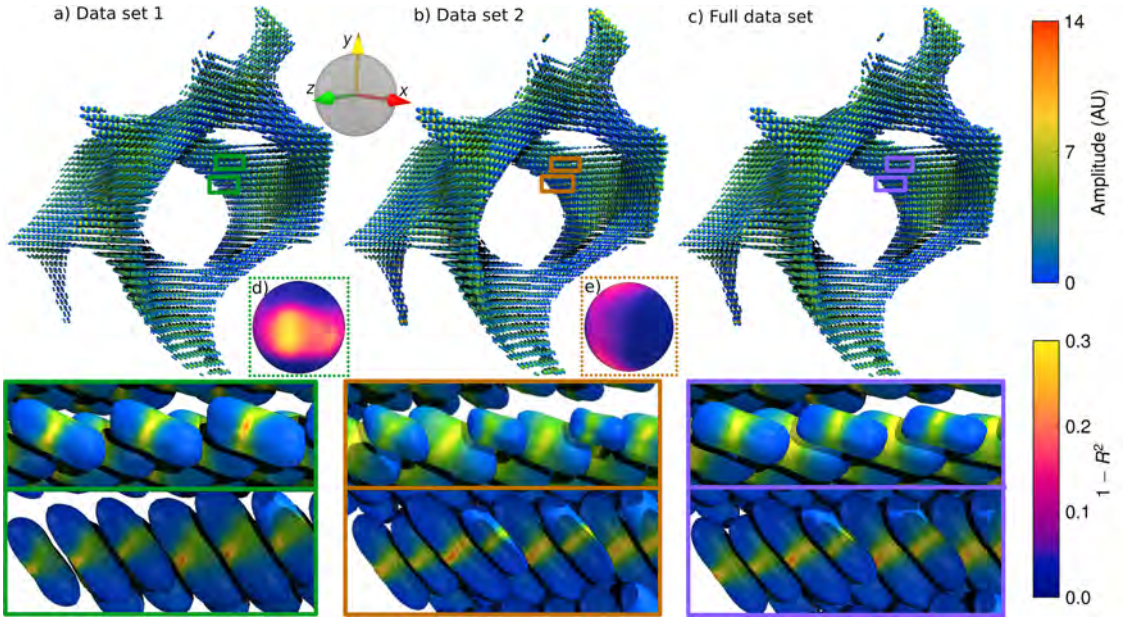

Figure 5.8: Spherical function glyph render of reconstructions. **a)** Partial data set 1, and a reciprocal space sphere showing the error distribution compared to the full data set. **b)** Partial data set 2. **c)** Full data set. **d)** Error distribution in reciprocal space for data set 1. **e)** Error distribution for data set 2. The color of each spherical function indicates the RSM amplitude, whereas the shape indicates the orientation. The shape is computed from the Funk-Radon transform of the RSM amplitude. The zoom-ins show two sets of RSMs that the partial reconstructions each have difficulty reconstructing, compared to the full data.

similar to those of the model in Liebi *et al.* (2015, 2018) [21, 22], could improve this issue in some instances.

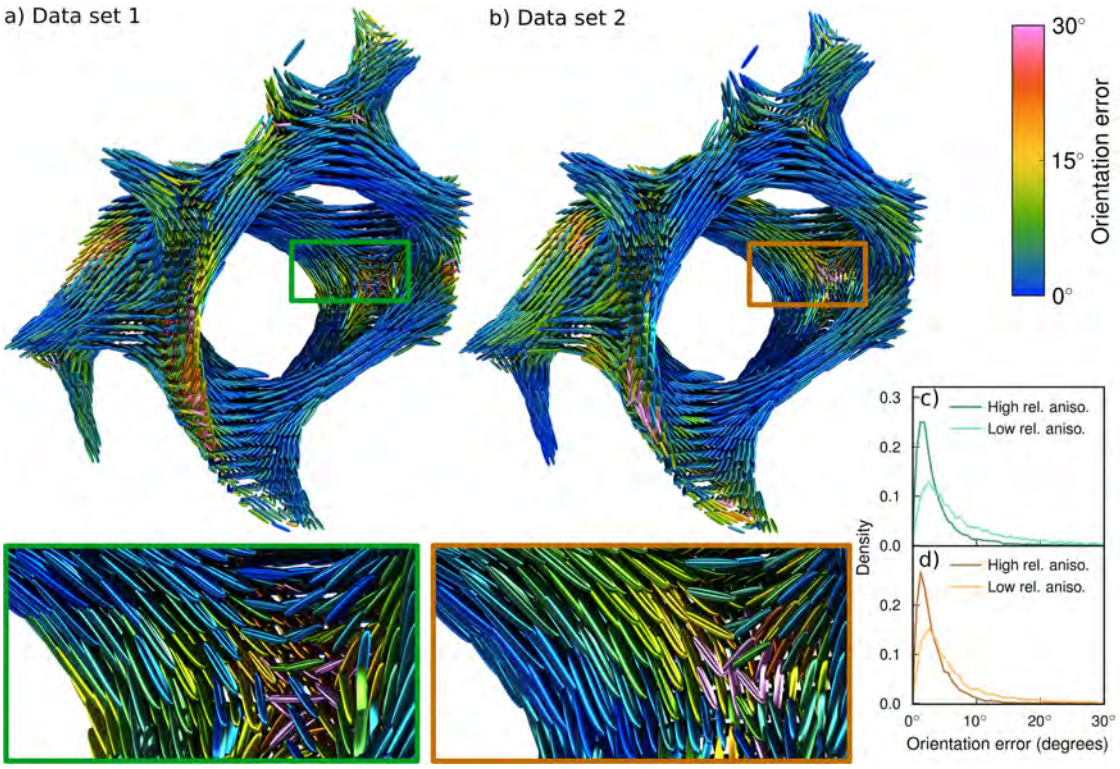

Figure 5.9: Orientation errors. a) Glyph render of orientations and errors of partial data set 1 **b)** Glyph render of data set 2 **c)** Probability density plot of orientation errors of partial data set 1. **d)** Probability density plot for partial data set 2. The color of each glyph indicates the orientation error in that voxel, with each glyph being scaled by the relative anisotropy in the partial reconstruction. The zoom-ins highlight an interface area where the orientation error is large. The density plots show the orientation errors for high (greater than 0.6) and low (less than 0.6) relative anisotropy, showing that the error is greater in low relative anisotropy regions.

6

Visualization techniques for tensor tomography

Visualization is an especially important aspect of any field of study that involves many-dimensional data. On a basic level, SAXSTT is the reconstruction of 6-dimensional fields. The three real-space dimensions have a natural interpretation – they map to the ordinary image coordinates in any three-dimensional image representation. The three reciprocal-space dimensions, however, must be treated with more consideration, because one cannot simply encode a three-dimensional field in every image coordinate, in a way that can be parsed by humans. There are in effect three approaches to dealing with this issue.

- First, one can extract an invariant or derived property of the reciprocal space fields – for example, a mean value, the orientation, a scalar parameter, or some combination of these, and plot this using volume or glyph rendering.
- Second, one can remove one dimension – *e.g.*, extract a single q or integrate over a q-range, and then plot a two-dimensional surface representing each RSM.
- Three, one can reduce the *real-space* dimensionality of the reconstruction and look at only a single RSM at a time. Then the RSM can be rendered using, *e.g.*, volume rendering.

In addition to covering these points, it is also necessary to discuss how we represent and conceptualize SAXSTT reconstructions, *i.e.*, what we consider one RSM in the reconstruction to represent.

6.1 Visualization representations

Generally speaking, three-dimensional visualization techniques require the definition of a *grid*, in addition to providing the data. A grid consists of *nodes*, and an implicit or explicit geometry describing how these nodes are connected. Node or point data specifies a value at one specific point. In addition, it may contain *cells*, which define volumes (or surfaces) between nodes. For example, a network of 4×4 equidistant nodes, connected in a rectilinear grid, would typically contain 3×3 cells. Cell data, unlike point data, specifies a value in a volume or area (which may be understood as an average value in that region).

Different softwares approach these issues differently, which is an easy source of off-by-one errors (*e.g.*, defining one node too many or too little). In this work, the primary software used for three-dimensional visualization is ParaView, which is a frontend for VTK [54]. While it may seem intuitive to treat voxel-by-voxel reconstructions as cell data, we will in fact treat it as node data. There are two reasons for this — the simpler reason is that node data is generally more efficient to work with than cell data. The more complicated reason is that per the discussion on the definition of the RSM in Sect. 2.1 and the John transform in Sect. 3.3, we do in fact to good approximation probe and reconstruct the RSM *at each point* within the volume. Thus, it is consistent to approach our data as node data, rather than as cell data.

6.2 Volume rendering

Volume rendering, see Fig. 6.1, is a common way to visualize scalar fields. It involves the integration of an *transfer function* over a field $f(\hat{\boldsymbol{r}})$, which may be written roughly,

$$S(\hat{\boldsymbol{r}} + \mathrm{d}\hat{\boldsymbol{p}}) - S(\hat{\boldsymbol{r}}) = T(S(\hat{\boldsymbol{r}}), f(\hat{\boldsymbol{r}} + \mathrm{d}\hat{\boldsymbol{p}}), \boldsymbol{\nabla} f(\hat{\boldsymbol{r}} + \mathrm{d}\hat{\boldsymbol{p}}), \ldots),$$

where T is some function of the probe S and the value and derivatives of the field f. The John transform, Algorithm 2, is an example of a very simple transfer function, which simply accumulates the value of the sample, and the resultant value can be mapped directly to, *e.g.*, a color. The idea behind a transfer function is that the probe passes through the sample at a given viewing direction, regularly samples it, and changes its value to reflect different regions of the sample. The transfer function maps different scalar values to different colors and opacities, and the final color of the probe at a given point then reflects the field viewed from that direction, giving an impression similar to that of looking through a semi-transparent object. For example, to visually emphasize a skeleton inside a body, soft tissue can be made semi-transparent, while the skeleton is made more opaque.

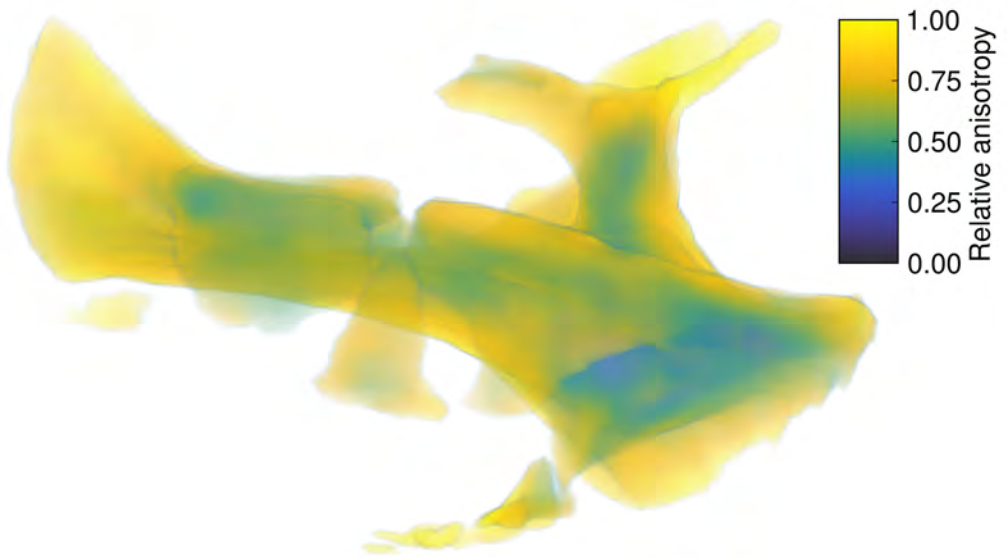

Figure 6.1: Volume render of trabecular bone, using the relative anisotropy as the scalar, after it has been thresholded to zero in areas where the mean falls below a pre-determined threshold. This is necessary, because the relative anisotropy, Eq. (5.2), goes to infinity as the mean goes to zero.

6.3 Glyph rendering

Glyphs are an alternative to volume rendering when we prefer to render discrete points, rather than a continuous volume. Each node in the volume is mapped to a glyph (a geometric shape, *e.g.*, a cylinder, arrow or box). In Fig. 6.2, we see an example of a *superquadric* glyph render, where the superquadric has been shaped into a slightly squared elongated pill shape. Each glyph can be individually scaled, colored and rotated, in order to encode multiple modes of information. Many frontends for, *e.g.*, VTK, such as ParaView, give only limited control of glyphs beyond this, even though the glyphs could in principle be scaled individually across each direction. Regardless of this, glyphs remain extremely useful for illustrating basic orientations of samples.

Flow-based orientation rendering options, such as line integral contours and streamlines, generally work best for relatively uniform orientation fields, since flows are unidirectional, while orientations are bidirectional (*i.e.*, there is no forward or reverse orientation). For relatively uniform fields, it is then possible to constrain the orientation vector to a particular hemisphere; an example of this is shown in Fig. 6.3, where the streamlines from the unrestricted orientation vectors in panel b) can be compared to the ones from hemisphere-restricted vectors in panel c).

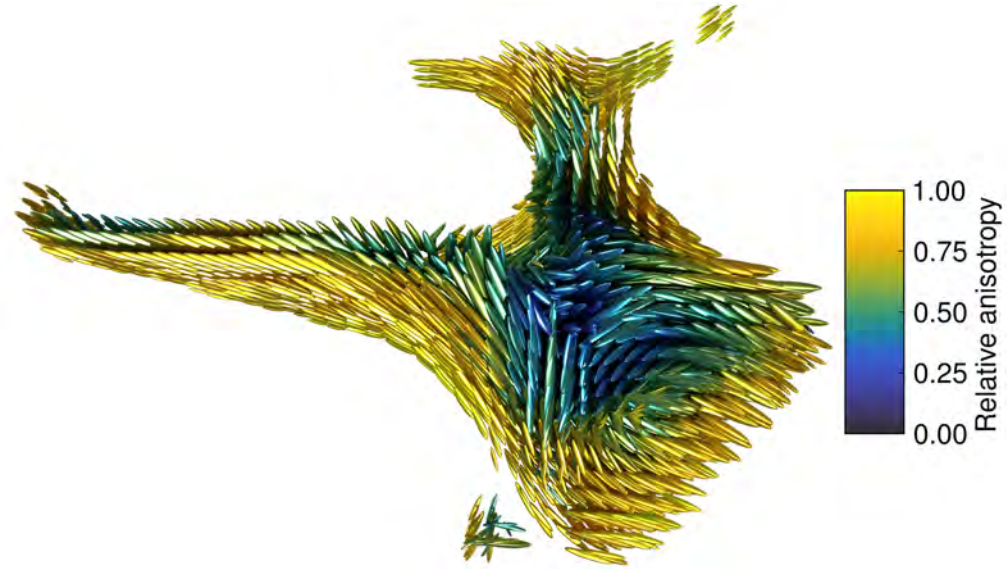

Figure 6.2: Superquadric glyph render of trabecular bone reconstruction, using the main orientation of each RSM to orient the glyphs, the mean amplitude to scale the glyphs, and the relative anisotropy, Eq. (5.2), as the color.

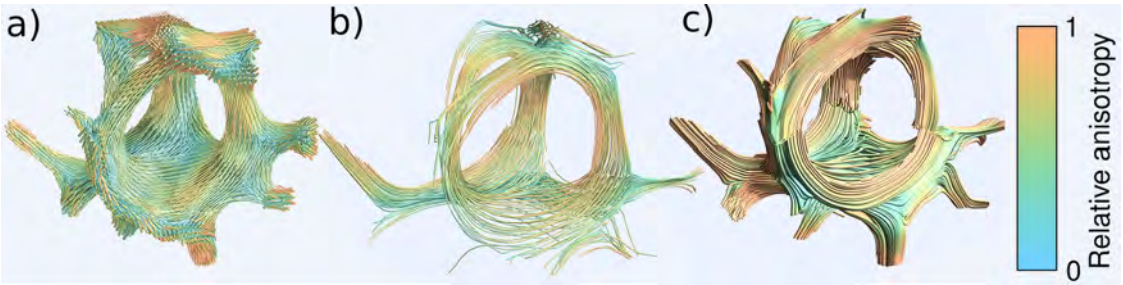

Figure 6.3: a) Cylinder glyph render showing orientation in trabecular bone. **b)** Streamline render. **c)** Streamline render after restricting orientations to one hemisphere, by restricting strict positivity of the orientation vector component orthogonal to the large loop.

6.4 Tensor field render

It is possible to use a tensor projected onto a spherical surface (or in the case of an RSM at a single q, to directly use that RSM) as a texture, and/or to morph the shape of a glyph, and thus directly render a tensor field. Within a framework such as VTK, glyphs filters are effectively a kind of Cartesian product of a glyph geometry and a node-valued field. In other words, each node in the field is replaced by a glyph geometry (with its own nodes and cells), and the values of the node are duplicated across every node in the glyph. This comes with both advantages and disadvantages. One of the basic problems of many such frameworks, including ParaView, is that they have no higher level of

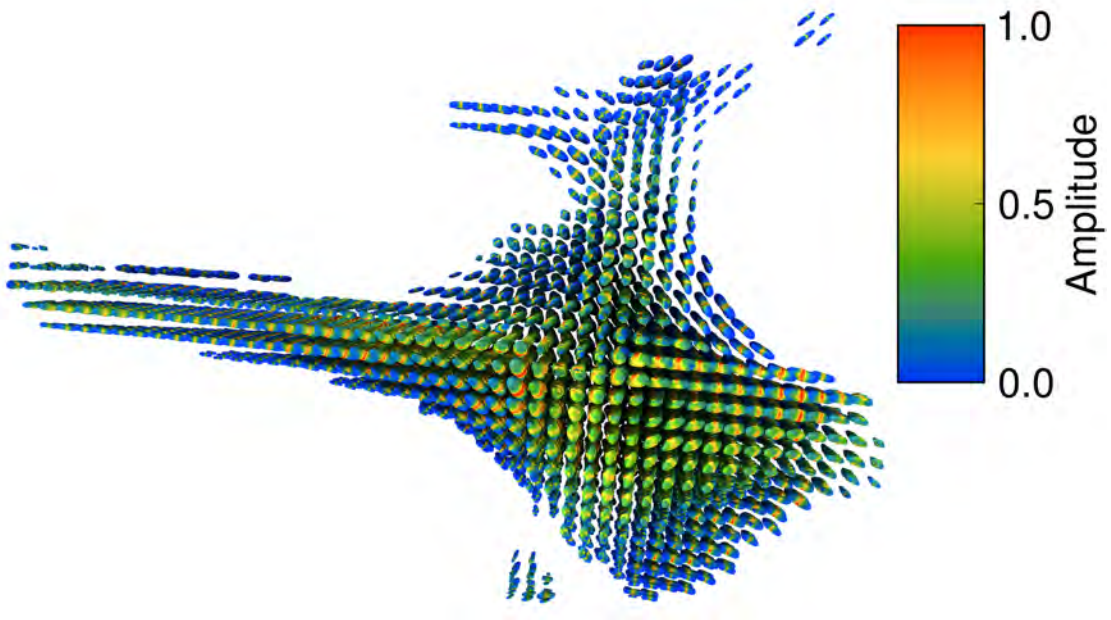

Figure 6.4: Tensor glyph render, using icosahedral spherical glyphs as the base glyphs, the Funk-Radon transform (Eq. (5.5)) of the RSM amplitude to shape the glyphs, and the RSM amplitude to color the glyph. The Funk-Radon transform is useful to ensure that the amplitude distribution is clearly visible, and in this case serves the skeuomorphic purpose of giving each tensor an elongated shape that indicates the directions of the fibers.

organization. That is, once the glyphs are created, they replace the original mesh, rather than being appended to it, requiring the duplication of values. The problem associated with this is that we cannot easily load a tensor field with hundreds of coefficients per voxel-node, then expand each voxel-node into a glyph, and then modify the glyphs to represent a tensor field, because the memory requirement associated with this is too great. We must instead go through a more involved process.

- Load necessary data for thresholding (*e.g.*, mean value of each RSM) at each node in the mesh.
- Threshold the mesh, removing all nodes which, *e.g.*, fail to meet a threshold mean amplitude.
- Take note of the coordinate at each node in the mesh, if not already recorded.
- Apply a spherical glyph filter, preferably using a relatively uniform spherical mesh, *e.g.*, one based on icosahedra.
- Record every coordinate in the new mesh.

- Sort the mesh so that it can be grouped into two dimensions, $[N, M]$ for N nodes in the original mesh, and M nodes in the spherical glyph mesh.

- Compute the reciprocal space projection matrix from every basis function to every point in the spherical glyph mesh.

- For each of the N node coordinates, load the corresponding RSM from the reconstruction, and compute the projection of the amplitude onto the M nodes in the spherical mesh, and store in an array of size $[N, M]$.

- Modify the coordinates of each point in the glyph-filtered mesh by subtracting the original node coordinates, multiplying the difference by the (normalized or re-scaled) projected amplitudes, and add the original node coordinates back. The result should be glyphs which have been morphed by the RSM amplitudes. Finally, use the amplitudes to texture the glyphs.

- Adapt as needed, by, *e.g.*, computing the Funk-Radon transform prior to morphing the glyphs, or by only using the amplitudes as texture rather than shape.

In Fig. 6.5, we can see an example of how the tensor glyph filter can be implemented in ParaView, using a programmable filter. This approach cannot necessarily be seamlessly copied to other visualization software, as it depends on range of assumptions about how the data is structured inside ParaView (*e.g.*, that one can reshape the glyph filtered mesh into a contiguous row-major matrix with N rows, and M columns, where N is the number of nodes in the thresholded mesh, and M is the number of nodes in each glyph). Thus, any adaptation of this approach needs to be complemented with commensurate knowledge about the framework to which it is adapted.

6.4.1 Rendering individual reciprocal space map

In addition to the rendering of three-dimensional fields with primarily real-space geometry, it is interesting to consider how to render individual three-dimensional reciprocal space maps. A single reciprocal space map is a function of $\boldsymbol{q}$, and it is natural to consider it in spherical coordinates, since a single value of $\|q\|$ corresponds to a single length scale. Thus, one relatively straight-forward way in which reciprocal space maps can be represented is on a structured curvilinear grid, which is locally homeomorphic to a cubic grid (*i.e.*, there is an invertible isomorphism between a cubic grid and the curvilinear grid, except at the poles). In ParaView, such a *structured grid* can be implicitly defined using a so-called custom source, and carrying out the following steps:

```
import numpy as np
from mumott.methods.basis_sets import GaussianKernels
from mumott.core.probed_coordinates import ProbedCoordinates
import h5py

source_size = 2562  # your source size
inp = inputs[0]  # Paraview input node
nodes = inp.PointData['grid'] # Original node coordinates
q_dir = inp.PointData['PointLocations'] - nodes  # Subtract node coordinates
mesh_size = q_dir.shape[0] // source_size  # get number of points in mesh
one_glyph = q_dir[:source_size]  # All glyphs have the same mesh
pc = ProbedCoordinates(vector=one_glyph.reshape(1, -1, 1, 3))
g = GaussianKernels(grid_scale=5, probed_coordinates=pc)  # Basis set

coefficients = np.zeros((geo_size, len(g)))
stride = nodes.shape[0] // (geo_size)

# Load coefficients, one-by-one
with h5py.File(f'my_reconstruction.h5') as f:
    c = f['basis_set/coefficients'][...]
    for i in range(mesh_size):
        p = tuple(nodes[i * stride].astype(int))  # Ravel index
        coefficients[i] = c[p]

amplitudes = g.get_amplitudes(coefficients, probed_coordinates=pc)

new_coordinates = q_dir * amps.reshape(-1, 1, 1) + nodes

output.SetPoints(new_coordinates.reshape(-1, 3))
output.PointData.append(amplitudes.ravel(), 'amplitudes')
```

Figure 6.5: Example of code necessary to do a tensor render in ParaView, here using a Python programmable filter.

1. Choose an angular resolution, *e.g.*, 32 azimuthal points and 16 polar points. Add 1 to the number of azimuthal points to account for the degeneracy $f(\theta, \phi) = f(\theta, \phi + 2\pi)$; an additional point at 2π is needed to make the field continuous[1].

2. Load all values of individual q-bins. Decide how to space each radial shell relative to each q-bin. You may wish to use logarithmic spacing, or to multiply the q-spacing by a constant so that the spacing is closer to 1 for, *e.g.*, camera manipulation purposes. In this work, I have used simple linear spacing in nm^{-1}, multiplied by a constant 75, yielding a maximum span of 240 length units in ParaView.

3. Enumerate each coordinate in a nested manner. For example, for each q, enumerate every θ, and for each enumerated θ, enumerate every ϕ. The order of enumeration is not important, as long as you are consistent about it. Compute the *Cartesian* coordinates corresponding to (q_a, θ_b, ϕ_c) and append sequentially to a list.

[1] It can even be advantageous to add more than one extra point, for the purpose of computing normal vectors, if the goal is to render surfaces.

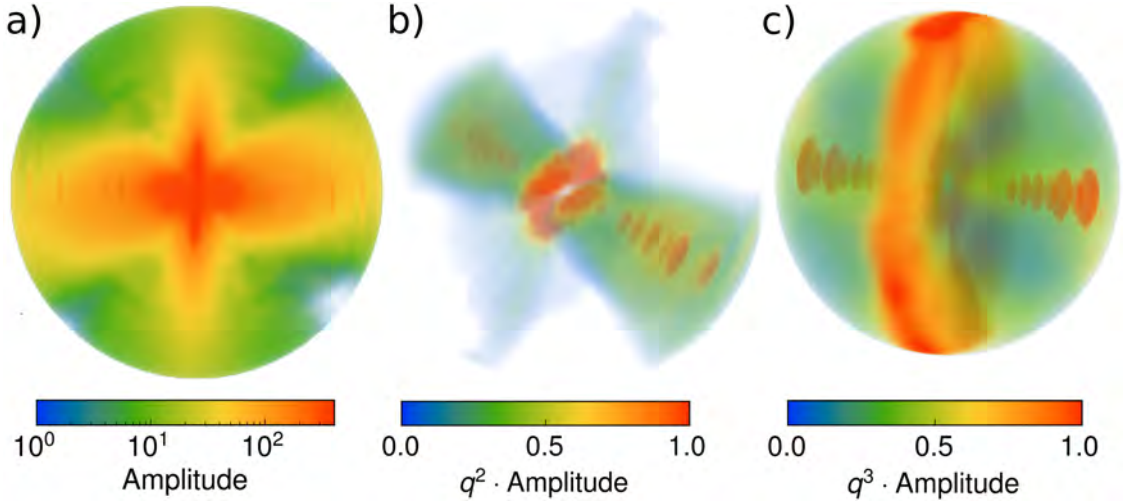

Figure 6.6: Volume renders of a RSM from highly oriented collagen-rich region of chameleon tongue. **a)** shows a volume render of the logarithm of the RSM amplitude, similar to how detector images are typically shown. The RSM has been cut along the middle and is viewed from the bottom of the hemisphere, in order to improve clarity. **b)** shows a Kratky plot-like render, scaling the amplitude by q^2, which is commonly done for SAXS data applied to biological systems. This makes it easier to see higher-q parts of the RSM. **c)** shows the amplitude scaled by q^3 instead, which emphasizes high-q regions even more. Each of the plots has advantages and disadvantages.

4. Specify the dimensions of the grid as (N_ϕ, N_θ, N_q) (assuming that ϕ is the innermost coordinate of your enumeration). Given that you have specified a Structured Grid and supplied the necessary metadata [2], ParaView/VTK will now automatically infer the structure - each cell will be a distorted cube, consisting of 8 nodes. This connectivity allows the visualization backend to make the proper inferences and interpolations which make it possible to apply further visualization steps.

The structured grid cannot be rendered directly, but it can be sliced and resampled to images. For example, this is the way in which the Ewald sphere cut illustration, Fig. 2.1 in Chapter 2, was made. We see three ways to volume-render a RSM in Fig. 6.6. Render a) is the most immediately familiar, being similar to a detector image, but also the most difficult to discern higher-q components in. In b) we see a Kratky-style render, which shows medium- and high-q components more clearly. Finally in c), which scales the amplitude by q^3, we see high-q components very clearly, but at the expense of lower-q components.

[2] Ensuring that the correct metadata is provided by a custom source in ParaView is highly technical and likely to change in future versions; the reader is thus referred to the VTK user's guide for details on this.

7

Using SAXSTT to study the chameleon tongue

The chameleon tongue is a very interesting biological system, due to the extremely high acceleration and velocity that the tongue undergoes when the chameleon catches insects. In fact, it was believed in antiquity that the chameleon fed on air, perhaps due to the high velocity of the tongue, making it hard to see with the naked eye [55]. The forces that must be exerted on the tongue in order to undergo extreme acceleration defy

Figure 7.1: Transmission tomography of the chameleon head. The arrow indicates the approximate region from where the SAXSTT sample was taken.

conventional biomechanical explanations, such as simple muscle action. Instead, the tongue must be looked at as a complex biomechanical system consisting of multiple interacting parts. For this reason, it is an excellent case study to demonstrate the power of SAXSTT as a "bridging" method, probing multiple scales at a time. In Paper III, we study the chameleon tongue using multiple modalities, and in this section we will look in-depth at the SAXSTT investigation of a section of the tip of the chameleon tongue, shown relative to a chameleon head in Fig. 7.1.

7.1 Reconstruction

The chameleon tongue presents a particularly challenging reconstruction case, for several reasons:

- The samples are quite large, at approximately one million voxels each.
- There are multiple tissue regions within each sample – collagen, fibro-cartilage, mineralized cartilage, and hyalocartilage, meaning that the reconstruction needs to accomodate a range of symmetries and a large amount of variability in both amplitude and texture.
- The collagen layer is extremely well aligned. This leads to very sharp scattering from the fibril diameter. Due to limitations in the angular resolution of the measurement, this scattering is only clearly captured at one measurement angle. This means that the overall smoothness constraints on tensor tomography are not likely to be satisfied, making reconstruction difficult.
- The RSMs in the chameleon tongue do not appear to have simple rotational symmetries. Some parts of the tongue show indications of multiple orientations, which makes it unsuited for reconstruction with basis functions that depend on a single orientation, such as the ZH model in Liebi *et al.* (2015, 2018) [21, 22].

These problems were alleviated by a range of actions. The diffraction images were integrated with relatively high resolutions (16 segments per half-circle), and a very high resolution in the reconstructed reciprocal space map was used (578 Gaussian kernel functions), in order to effectively capture a wide range of textures. The reconstruction was carried out using a relatively large number of iterations (100 per q-bin) and a reduced step size, along with two regularizers, for the coefficient-wise total variation and for the L_1 norm. Due to the very sharp scattering in certain areas, the reconstruction was also deliberately thresholded so that no coefficient had a lower value than 0. This is because the reconstruction may otherwise attempt to make certain coefficients negative in order to allow for a sharper increase, leading to overfitting. In total, the reconstruction of one sample took approximately 10 minutes per q-bin, using the asynchronous reconstruction

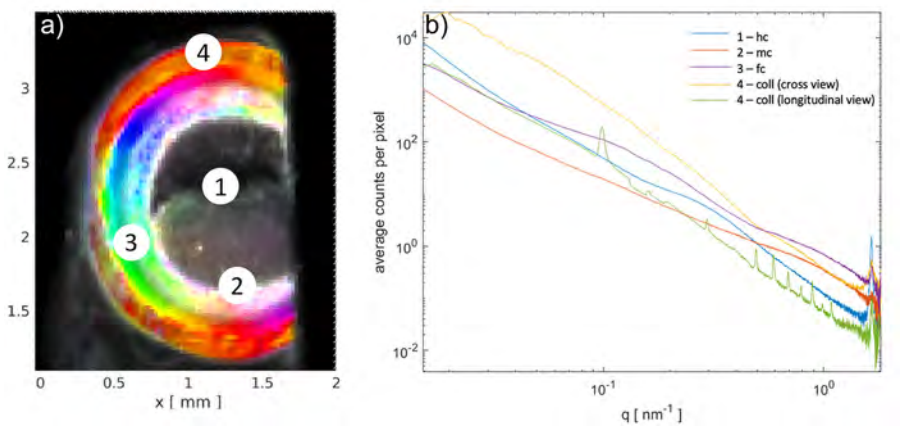

Figure 7.2: a) Measured two-dimensional projection of the SAXSTT sample with orientation analysis performed between 0.273 – 0.314 nm^{-1}. This shows the four primary layers: **1** – hyaline cartilage (hc), **3** – mineralised cartilage (mc), **3** – fibro-cartilage (fc), and **4** – collagen layer (coll). **b)** SAXS curves from the different layers. The collagen layer in longitudinal view is taken from the sample rotated 90 degrees about the y-axis, where the d-spacing of the collagen gap is in Bragg conditions for the collagen aligned in the longitudinal direction.

pipeline MOTR in MUMOTT, or approximately 32 hours across all 194 $\|\boldsymbol{q}\|$-bins, using an Nvidia A100 GPU.

7.2 q-resolved analysis

The reconstruction of the chameleon tongue across 194 q-bins allows different tissues to be differentiated based on their characteristic scattering signals. The most prominent one is thereby collagen typ I, as it exhibits distinct diffraction peaks (see Fig. 7.2, green curve) from the supramolecular staggered structure, with a periodicity of about 67 nm along the fiber axis, *i.e.*, in meridional direction, while exhibiting scattering from the packing of the collagen fibrils in the equatorial direction [56]. We know from histology (paper III) that other tissue types present are different types of cartilage, *i.e.*, hyalo-cartilage, fibrocartilage and mineralized cartilage. The scattering signature of those tissue types show less distinct scattering features as shown in Fig. 7.2. The power-law in a low to intermediate q-range 0.13 – 0.2 nm^{-1}, can be used as a general-purpose tool to differentiate between tissues with different scattering characteristics. In the SAXS theory of isolated particles, the slope in the log I vs. log q curve is related to the fractal dimension. For example, a slope of -2 corresponds to a disk-shape, and -1 corresponds to a rod-shape. However, the interpretation is less clear in solid complex tissues, where form- and structure factor of several nanostructures contribute to the total scattering. Thus, the interpretation is not necessarily clear, but the slope can still be

used to identify different tissue types. It is thereby a valuable quantity for segmentation. In the scattering analysis of bone, this power-law is often referred to as the G-exponent. It is used to identify regions where so called T-parameter analysis can be applied to to estimate the thickness of the mineralized platelets. In a SAXS curve of bone, the hydroxy-apatite mineral signature can be seen in a shoulder appearing at high-q values [4]. This signature is weakly visible in the selected point from one projection through the sample (Fig. 7.2, orange curve). The T-parameter analysis assumes a two-phase system with a mineral fraction of 50 % with predominantly platelet-shaped particles and has also been used to characterize mineralized cartilage [57]. While it is unclear whether these assumptions hold for the mineralized cartilage of the chameleon tongue, we use this analysis to investigate how uniform the mineral component in the layer is. The T-parameter analysis includes the determination of the scattering invariant from the Krakty plot, which is only possible when the slope of the scattering signal at low-q is lower or equal to 2. Consequently, this the analysis is only performed where the G-exponent is ≤ 2. The G-exponent was computed through a fit, using the power-law approximation [58]

$$\mathrm{RSM}(\boldsymbol{r}, \boldsymbol{q}) \approx a(\boldsymbol{r}) \, \|\boldsymbol{q}\|^{-G(\boldsymbol{r})},$$

in the q-range $0.13 - 0.20\,\mathrm{nm}^{-1}$ and the T-parameter using the integral [59]

$$T(\boldsymbol{r}) = \frac{r}{\pi P} \int_0^\infty \mathrm{d}q \, \|\boldsymbol{q}\|^2 \, \mathrm{RSM}(\boldsymbol{r}, \|q\|),$$

in the q-range $0.2 - 1.2\,\mathrm{nm}^{-1}$, and where where $\boldsymbol{P}$ is the Porod constant, which was determined from the q-range $0.77 - 1.2\,\mathrm{nm}^{-1}$. These scalar parameters have previously been studied in SAXSTT reconstructions by Liebi *et al.* (2021) and Casanova *et al.* (2023) [60, 61]. The T-parameter was only computed from the mineralized areas, which were identified based on the Porod constant.

The computed G-exponent and T-parameter can be seen in Fig. 7.3.

The chameleon tongue reconstruction was segmented into 4 components, aided by tissue characterization through histology, scanning SAXS, and birefringence microscopy (see Paper III). The components, shown as a telescope figure in Fig. 7.4a), were identified as an outer collagen sheath, a fibrocartilage layer, a mineralized tissue layer, and an innermost hyalocartilage layer.

The collagen layer was identified by its large G-exponent and extremely sharp scattering in both the equatorial (from the packing of the collagen fibrils) and meridional directions (from the staggered spacing of collagen fibers). The mineralized layer was identified by its Porod constant, and the fibrocartilage layer was then found as the part between the mineralized layer and the collagen layer. Finally the hyalocartilage layer was the remaining, weakly scattering part.

It was observed that the collagen layer and the fibrocartilage layer did not separate completely, and that there appeared to be an interface layer between them, which can

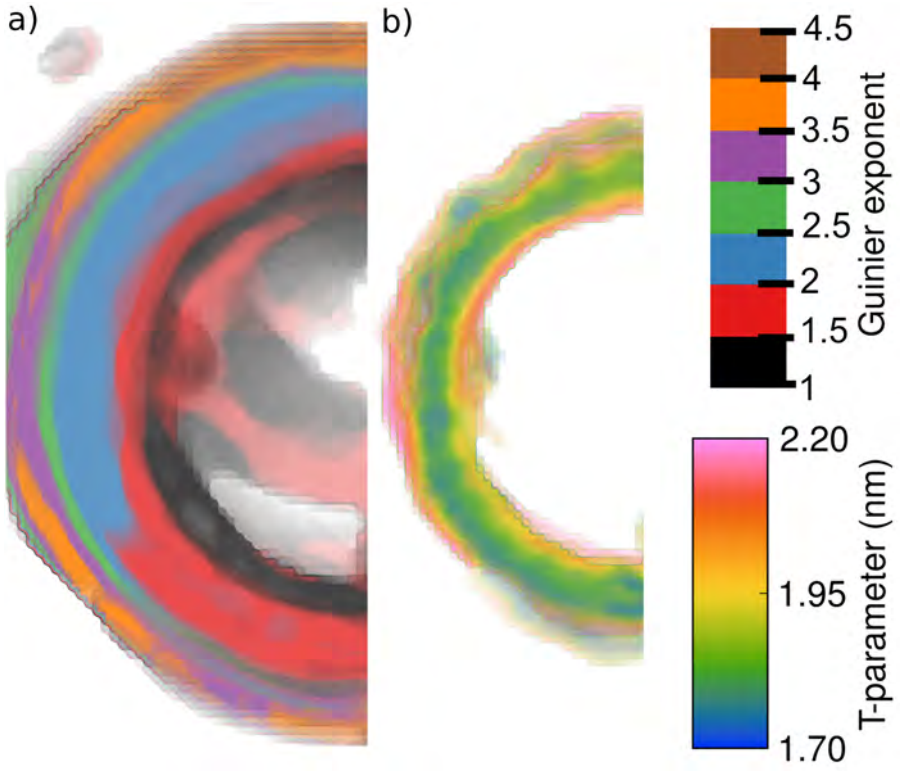

Figure 7.3: a) G-exponent and **b)** T-parameter of chameleon tongue. The T-parameter is computed from the part of the sample where the G-exponent is no greater than 2.

also be seen as a narrow green band between the thicker blue and orange-purple bands in the G-exponent plot, Fig. 7.3a).

Fig. 7.5 shows volume renders of RSMs from four regions, viewed from a central cut orthogonal to the viewing direction. The horizontal direction is the long axis of the chameleon tongue, whereas the vertical direction is the main tomographic axis. We observe in Fig. 7.5b) (interface) and to a lesser extent in c) (fibro-cartillage) meridional collagen peaks both in the horizontal and vertical direction. This indicates that each RSM in the interface region has multiple orientations. However, because the collagen meridional scattering occurs in the lower-quality direction (see Sect. 5.3), 90° from the main tomographic axis, it is difficult to say to what extent this reflects a real feature in the sample.

More traditional ways of illustrating the RSM are shown in Fig. 7.6. In panel a) a standard log-log plot of $I(q)$ as a function of q is shown for a mineralized RSM, along with the three regions used for the T-parameter analysis. In panel b), a Kratky plot of the same area is shown. Panels c)-e) show the three characteristic q-ranges used for the segmentated render in Fig. 7.4, and the locations of the four RSMs shown in Fig. 7.5.

In Fig. 7.7 we see the same views as in Fig. 7.5, but the amplitudes are scaled by $\|\boldsymbol{q}\|_2^2$,

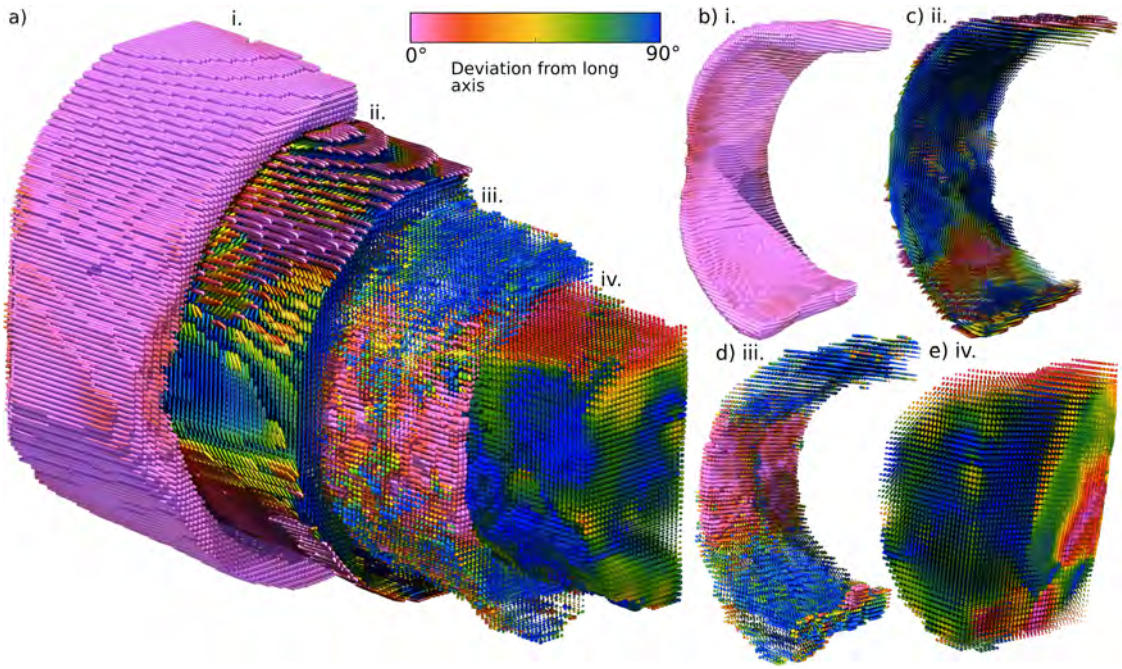

Figure 7.4: Telescope view of chameleon tongue reconstruction. a) The four principal layers of the tongue section, labelled **i** through **iv**. **b)** Collagen layer ($q = 0.095\,\mathrm{nm}^{-1}$). **c)** Fibrocartilage layer ($q = 0.126\,\mathrm{nm}^{-1}$). **d)** Mineralized cartilage layer ($q = 0.733\,\mathrm{nm}^{-1}$). **e)** Hyalocartilage layer ($q = 0.041\,\mathrm{nm}^{-1}$). The glyphs are oriented according to the principal orientation of the reciprocal space map in each voxel, scaled according to the relative anisotropy in characteristic q-ranges. They are coloured according to their deviation from the long axis of the tongue. The collagen layer **i**. and the fibrocartilage layer **ii**. do not separate completely in the segmentation, which is consistent with the existence of an interface layer.

as in a Kratky plot, which is used in the characterization of biological systems [62]. Here we employ it to improve contrast in the rendering of the RSM. We observe in Fig. 7.5b) (interface) and to a lesser extent in c) (fibro-cartillage) meridional collagen peaks both in the horizontal and vertical direction. The RSM amplitude which results from the fiber diameter (vertical) in a) can be seen to be extremely sharp, which causes difficulties during reconstruction as discussed in Sect. 7.1. Note also that the orthogonality of the orientation of the amplitude minimum in c) and d) is clear in this view.

7.3 Multi-orientation analysis

One of the advantages of the complex texture reconstruction introduced in Paper I is that it enables the retrieval of complex features of the RSM which would not be possible to reconstruct with symmetry-constrained or lower-order models. However, it is necessary

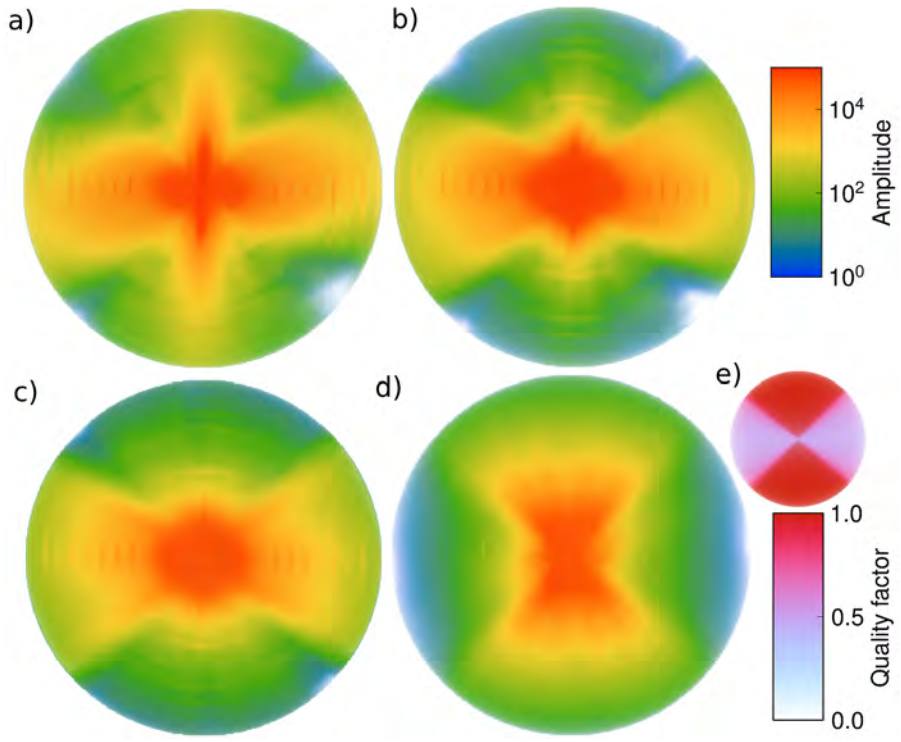

Figure 7.5: Volume renders of RSMs from **a)** the main collagen layer, **b)**, the interface layer between collagen and fibrocartilage, **c)**, the fibrocartilage layer, and **c)** the mineralized cartilage layer. **e)** shows the missing-wedge quality factor given by Eq. 5.6. The color and opacity transfer functions of the RSMs are individually adjusted to reveal their individual features, but all are shown using a logarithmic scale.

to take other reconstruction limitations into account, principally the missing wedge problem, as discussed in Paper II and Sect. 5.3. With this in mind, we must interpret the indications of multiple orientations seen in *e.g.*, Fig. 7.7b) with caution.

The collagen-fibrocartilage interface was extracted by finding a contiguous layer with values of the G-exponent and relative anisotropy between those of the collagen and fibrocartilage layers. This layer appears to have multiple orientations, and an analysis was carried out to extract these orientations. For this analysis, we selected the 5th reflection of the collagen meridional peak, in the q-range $0.536 - 0.677\,\mathrm{nm}^{-1}$. The meridional peaks were extracted by computing the difference integral over this peak

$$M(\boldsymbol{r}) = \int_a^b \mathrm{d}q \left[\mathrm{RSM}'(\boldsymbol{r}, \boldsymbol{q}) - u(\boldsymbol{r}, \hat{\boldsymbol{q}}) \, \|\boldsymbol{q}\|^{w(\boldsymbol{r}, \hat{\boldsymbol{q}})} \right],$$

where RSM′ is the RSM after subjecting it to convolution by a low-pass filter, in order to reduce the extraction of spurious peaks due to, *e.g.*, noise. The intercept $u(\boldsymbol{r})$ and exponent $w(\boldsymbol{r}, \hat{\boldsymbol{q}})$ are extracted by fitting a power-law to the end-points $\mathrm{RSM}'(\boldsymbol{r}, a\hat{\boldsymbol{q}})$ and

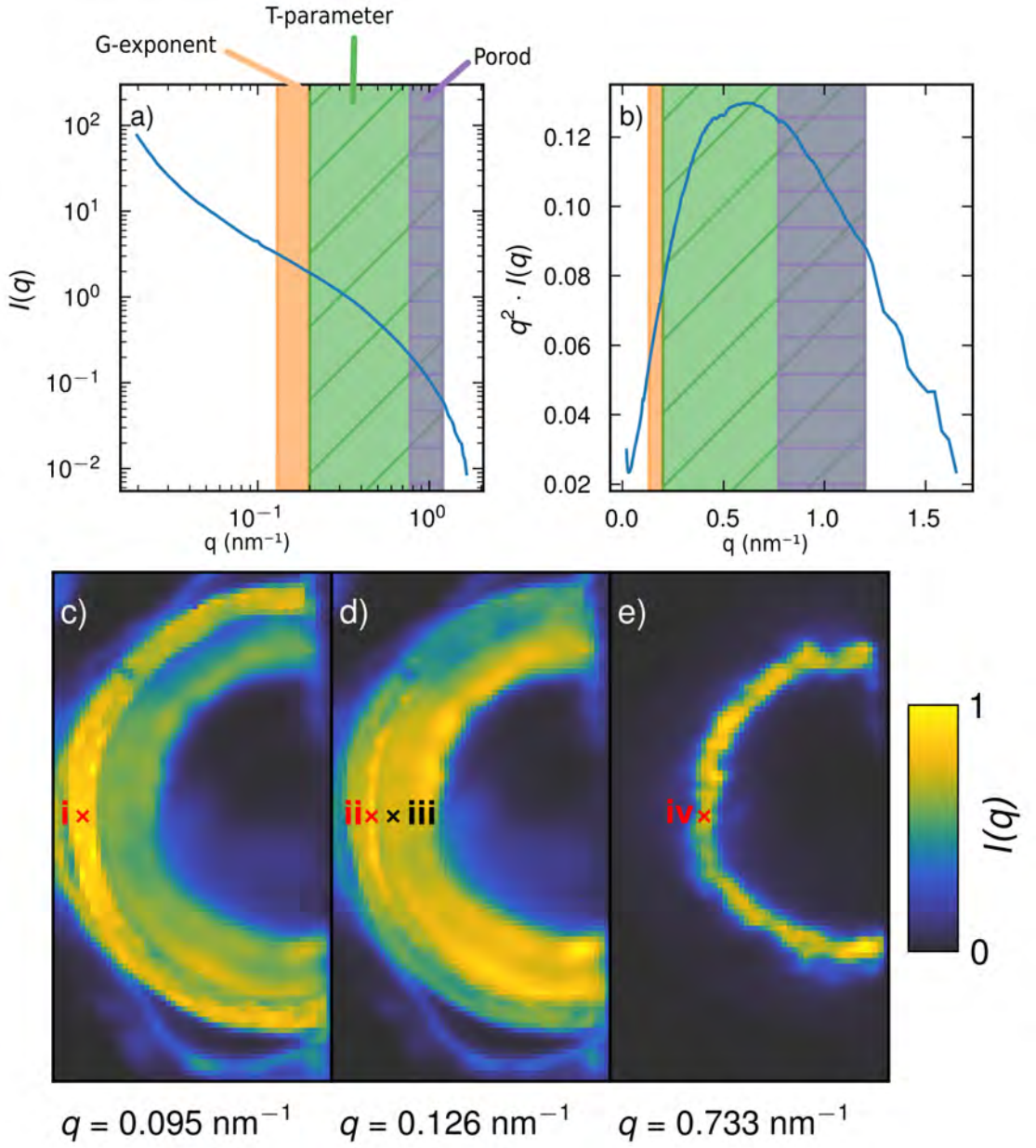

Figure 7.6: **a)** Mean RSM amplitude for mineralized part of tissue, with Guinier (orange), T-parameter (green), and Porod (purple) regions highlighted. **b)** Same curve, but in a Kratky plot which suggests the underlying bell-shaped curve which is integrated over to yield the T-parameter. **c)** Mean scattering at collagen characteristic q, with the location of the collagen RSM in Fig. 7.5a) annotated as **i**. **d)** Fibrocartillage characteristic q, with the interface and fibrocartillage RSMs in Fig. 7.5b) and c) identified as **ii** and **iii** respectively. **e)** Mineral characteristic q, with the RSMs in Fig. 7.5d) identified as **iv**.

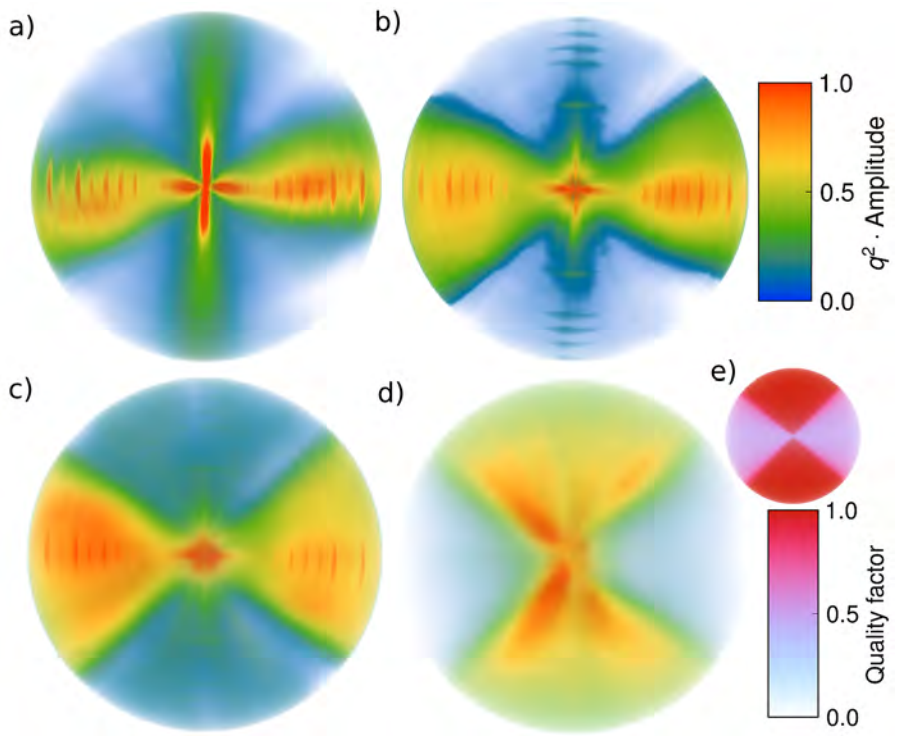

Figure 7.7: Kratky-rescaled RSMs from **a)** the main collagen layer, **b)**, the interface layer between collagen and fibrocartilage, **c)**, the fibrocartilage layer, and **c)** the mineralized cartilage layer. **e)** shows the missing-wedge quality factor given by Eq. 5.6. The color and opacity transfer functions of the RSMs are individually adjusted to reveal their individual features, but all are shown on a linear map.

$\text{RSM}'(\boldsymbol{r}, b\hat{\boldsymbol{q}})$. The result of the multiply-oriented collagen peak extraction can be seen in Fig. 7.8. It appears that there is a very consistent network of orientations aligned with both the long axis of the tongue as well as with the tangential direction (wrapping around the axis). The consistency we observe in the network within the well defined interface layer, in the same position where we found an interface layer with scanning SAXS and birefringence (see Paper III), makes us confident that even accounting for the limitations indicated by the quality factor of the RSM, which are introduced by the missing wedge problem, the extracted multiple orientations reflect real structural features.

In Fig. 7.9, an illustration of the integrals over, respectively, RSM' and $\|\boldsymbol{q}\|^w$ is shown. The amplitude of the reciprocal space map was integrated over the 5th collagen peak, and the integral of a power law fitted under this peak was subtracted. Note that the integrated RSM have considerable amounts of amplitude going in the radial direction, although there are no apparent spurious peaks in this direction, suggesting the extraction procedure is relatively robust.

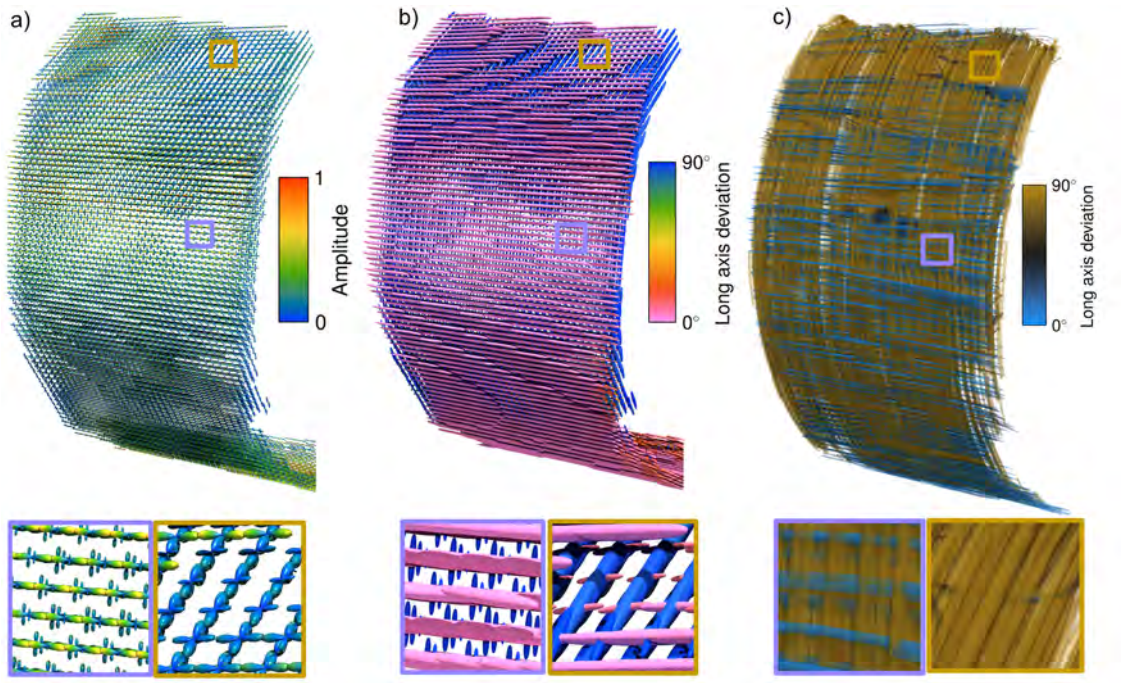

Figure 7.8: Interface layer with multiple orientations. a) Tensor glyph renders of extracted collagen meridional peak reciprocal space map. **b)** Multiple orientations extracted from collagen peak shown using oriented superquadric glyphs, scaled by the relative amplitude of the extracted peaks in each direction. **c)** Multiple orientations shown using streamlines.

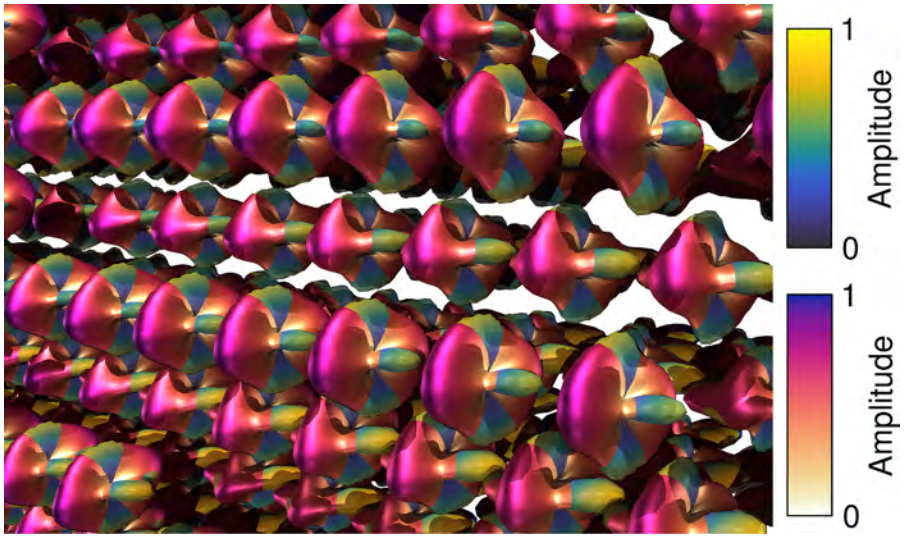

Figure 7.9: Detail showing scaled and colored by the amplitude of the power-law integral under the collagen peak (solid, white-to-blue, metallic) and the total integral including the peak (translucent, dark-blue-to-yellow). The integral including the peak is consistently slightly larger in two directions, which is where the extracted peaks in Fig. 7.8 comes from.

8

Conclusions and outlook

This work has detailed advances in the theoretical characterization and scope of SAXSTT as a method, as well as improvements in the speed and usability of reconstruction implementations. The algorithms and optimization theory underlying these improvements, and their implications, have been described and discussed in detail. The robustness and extent of validity of the reconstruction methods implemented in MUMOTT have been demonstrated using both simulated and experimental data. Furthermore, these advances have been leveraged in a case study of SAXSTT applied to a chameleon tongue sample, where multiple orientation analysis illustrates the advantages of parameterizing the RSM as a general function on the sphere.

The improvements to reconstruction speed, quality and robustness that have been obtained for SAXSTT within the scope of this work open up for commensurate improvements in data acquisition. Not only do the many orders of magnitude of improvements in speed while maintaining support for complex RSM textures enable larger and more detailed samples to be reconstructed, the improvements in robustness and better theoretical understanding open up for further development in sparse acquisition schemes. It remains to be investigated to what extent the amount of acquired data can be reduced while maintaining a sufficiently high quality of the reconstructions. The use of suitable regularization terms and representations for the RSM is likely to be key in this investigation.

A topic related to sparse acquisition is the possibility of online reconstruction, where the SAXSTT reconstruction is carried out simultaneously with the acquisition of data. The acquisition could then be guided by the reconstruction and terminated when the reconstruction has converged. An example of data-driven sampling would be for the acquisition to adapt to regions with very sharp scattering, such as the equatorial fiber scattering from collagen fibrils in the chameleon tongue (see, *e.g.*, Fig. 7.5), and measure additional directions adjacent to those of the sharp scattering maximum.

While not directly related to algorithmic improvement or performance optimization, the visualization of the reconstructed 5- or 6-dimensional fields is an interesting avenue for research. It is difficult to combine, *e.g.*, a scalar volume render with oriented glyphs into a render that is clear and visually appealing, and translucent surfaces are difficult to render well even with ray tracing. However, it is likely possible to not only write implementations specifically for showing combined glyph-and-volume renders, but possibly also for directly rendering SAXSTT reconstructions. For example, streamlines could be generated based directly on the RSM (as a function of the unit sphere), rather than derived vectors, and the render could be textured based on how much streamlines diverge, intersect, and so forth. Realizing this and similar ideas would likely require specialized expertise in visualization. This would greatly improve and accelerate the interpretation of results from SAXSTT.

Further improvements to the structure of MUMOTT could enable users to take advantage of the speed gains of asynchronous reconstructions, while preserving the modularity and interactivity of an object-oriented framework. While this requires careful implementation and thorough testing, it is well within the capabilities of Numba-CUDA and could greatly improve the user experience. Moreover, the adoption of hardware-agnostic frameworks like SyCL could enable the use of GPUs from vendors besides Nvidia. Another area with room for improvement concerns the implementation of explicit support for q-resolved reconstruction, including regularization or preconditioning which uses neighbouring q-bin reconstructions to accelerate and otherwise improve reconstruction.

Finally, the intended scope of MUMOTT stretches beyond any individual modality, such as SAXS. Future development should consider not only how to improve support for SAXSTT as a method, but also how aspects of the SAXSTT implementation can be re-used for, *e.g.*, wide-angle x-ray scattering tensor tomography (WAXSTT). Taking a wider perspective still, support for still more general modalities can be considered, such as 3D x-ray diffraction or dark-field diffraction grating tomography. This would include systems which can not necessarily be reduced to linear computed tomography problems, and which would require further improvements still to the implemented optimization algorithms.

Bibliography

[1] P. Fratzl, H. F. Jakob, S. Rinnerthaler, P. Roschger, and K. Klaushofer, *Position-Resolved Small-Angle X-ray Scattering of Complex Biological Materials,* Journal of Applied Crystallography **30**, 765 (1997). doi:10.1107/S0021889897001775.

[2] S. Pabisch, W. Wagermaier, T. Zander, C. Li, and P. Fratzl, *Chapter Eighteen - Imaging the Nanostructure of Bone and Dentin Through Small- and Wide-Angle X-Ray Scattering,* in *Research Methods in Biomineralization Science*, edited by J. J. De Yoreo (Academic Press, 2013), p. 391. doi:10.1016/B978-0-12-416617-2.00018-7.

[3] O. Paris, *From diffraction to imaging: New avenues in studying hierarchical biological tissues with x-ray microbeams (Review),* Biointerphases **3**, FB16 (2008). doi:10.1116/1.2955443.

[4] P. Fratzl, S. Schreiber, and K. Klaushofer, *Bone Mineralization as Studied by Small-Angle X-Ray Scattering,* Connective Tissue Research **34**, 247 (1996). doi:10.3109/03008209609005268.

[5] M. Georgiadis, R. Müller, and P. Schneider, *Techniques to assess bone ultrastructure organization: orientation and arrangement of mineralized collagen fibrils,* Journal of The Royal Society Interface **13**, 20160088 (2016). doi:10.1098/rsif.2016.0088.

[6] H. Lichtenegger, M. Müller, O. Paris, C. Riekel, and P. Fratzl, *Imaging of the helical arrangement of cellulose fibrils in wood by synchrotron X-ray microdiffraction,* Journal of Applied Crystallography **32**, 1127 (1999). doi:10.1107/S0021889899010961.

[7] J. Goldstein, D. Newbury, D. Joy, C. Lyman, P. Echlin, E. Lifshin, L. Sawyer, and J. Michael, *Scanning Electron Microscopy and X-ray Microanalysis ISBN: 0306472929* (, 2003). ISBN 0306472929. doi:10.1007/978-1-4615-0215-9.

[8] P. Thibault, M. Dierolf, A. Menzel, O. Bunk, C. David, and F. Pfeiffer, *High-Resolution Scanning X-ray Diffraction Microscopy,* Science **321**, 379 (2008). doi:10.1126/science.1158573.

[9] M. Dierolf, A. M. Menzel, P. Thibault, P. Schneider, C. M. Kewish, R. Wepf, O. Bunk, and F. Pfeiffer, *Ptychographic X-ray computed tomography at the nanoscale,* Nature **467**, 436 (2010). https://api.semanticscholar.org/CorpusID:2449015.

[10] M. K. Croughan, Y. Y. How, A. Pennings, and K. S. Morgan, *Directional dark-field retrieval with single-grid x-ray imaging,* Opt. Express **31**, 11578 (2023). doi:10.1364/OE.480031.

[11] F. Natterer and F. Wübbeling, *Mathematical Methods in Image Reconstruction* (Society for Industrial and Applied Mathematics, 2001). doi:10.1137/1.9780898718324.

[12] J. M. Feldkamp, M. Kuhlmann, S. V. Roth, A. Timmann, R. Gehrke, I. Shakhverdova, P. Paufler, S. K. Filatov, R. S. Bubnova, and C. G. Schroer, *Recent developments in tomographic small-angle X-ray scattering,* physica status solidi (a) **206**, 1723 (2009). doi:10.1002/pssa.200881615.

[13] C. G. Schroer, M. Kuhlmann, S. V. Roth, R. Gehrke, N. Stribeck, A. Almendarez-Camarillo, and B. Lengeler, *Mapping the local nanostructure inside a specimen by tomographic small-angle x-ray scattering,* Applied Physics Letters **88**, 164102 (2006). doi:10.1063/1.2196062.

[14] N. Stribeck, U. Nöchel, S. S. Funari, and T. Schubert, *Tensile tests of polypropylene monitored by SAXS. Comparing the stretch-hold technique to the dynamic technique,* Journal of Polymer Science Part B: Polymer Physics **46**, 721 (2008). doi:10.1002/polb.21403.

[15] R. Seidel, A. Gourrier, M. Kerschnitzki, M. Burghammer, P. Fratzl, H. S. Gupta, and W. Wagermaier, *Synchrotron 3D SAXS analysis of bone nanostructure,* Bioinspired, Biomimetic and Nanobiomaterials **1**, 123 (2012). doi:10.1680/bbn.11.00014.

[16] T. H. Jensen, M. Bech, O. Bunk, A. Menzel, A. Bouchet, G. Le Duc, R. Feidenhans'l, and F. Pfeiffer, *Molecular X-ray computed tomography of myelin in a rat brain,* NeuroImage **57**, 124 (2011). doi:10.1016/j.neuroimage.2011.04.013.

[17] M. Álvarez-Murga, P. Bleuet, and J.-L. Hodeau, *Diffraction/scattering computed tomography for three-dimensional characterization of multi-phase crystalline and amorphous materials,* Journal of Applied Crystallography **45**, 1109 (2012). doi:10.1107/S0021889812041039.

[18] P. De Falco, R. Weinkamer, W. Wagermaier, C. Li, T. Snow, N. J. Terrill, H. S. Gupta, P. Goyal, M. Stoll, P. Benner, and P. Fratzl, *Tomographic X-ray scattering based on invariant reconstruction: analysis of the 3D nanostructure of bovine bone,* Journal of Applied Crystallography **54**, 486 (2021). doi:10.1107/S1600576721000881.

[19] N. Stribeck, A. A. Camarillo, U. Nöchel, C. Schroer, M. Kuhlmann, S. V. Roth, R. Gehrke, and R. K. Bayer, *Volume-Resolved Nanostructure Survey of a Polymer Part by Means of SAXS Microtomography,* Macromolecular Chemistry and Physics **207**, 1139 (2006). doi:https://doi.org/10.1002/macp.200600147.

[20] F. Schaff, M. Bech, P. Zaslansky, C. Jud, M. Liebi, M. Guizar-Sicairos, and F. Pfeiffer, *Six-dimensional real and reciprocal space small-angle X-ray scattering tomography,* Nature **527**, 353 (2015). doi:10.1038/nature16060.

[21] M. Liebi, M. Georgiadis, J. Kohlbrecher, M. Holler, J. Raabe, I. Usov, A. Menzel, P. Schneider, O. Bunk, and M. Guizar-Sicairos, *Small-angle X-ray scattering tensor tomography: model of the three-dimensional reciprocal-space map, reconstruction algorithm and angular sampling requirements,* Acta Crystallographica Section A **74**, 12 (2018). doi:10.1107/S205327331701614X.

[22] M. Liebi, M. Georgiadis, A. Menzel, P. Schneider, J. Kohlbrecher, O. Bunk, and M. Guizar-Sicairos, *Nanostructure surveys of macroscopic specimens by small-angle scattering tensor tomography,* Nature **527**, 349 (2015).

[23] O. Glatter and O. Kratky, *Small angle x-ray scattering / edited by O. Glatter and O. Kratky.* (London ;: Academic Press, 1982).

[24] T. H. Jensen, M. Bech, O. Bunk, M. Thomsen, A. Menzel, A. Bouchet, G. L. Duc, R. Feidenhans'l, and F. Pfeiffer, *Brain tumor imaging using small-angle x-ray scattering tomography,* Physics in Medicine & Biology **56**, 1717 (2011). doi:10.1088/0031-9155/56/6/012.

[25] C. Kittel, *Introduction to Solid State Physics* (Wiley, 2004). ISBN 9780471415268.

[26] J. Driscoll and D. Healy, *Computing Fourier Transforms and Convolutions on the 2-Sphere,* Advances in Applied Mathematics **15**, 202 (1994). doi:https://doi.org/10.1006/aama.1994.1008.

[27] W. Dehnen and H. Aly, *Improving convergence in smoothed particle hydrodynamics simulations without pairing instability,* Monthly Notices of the Royal Astronomical Society **425**, 1068 (2012). doi:10.1111/j.1365-2966.2012.21439.x.

[28] C. Shannon, *Communication in the Presence of Noise,* Proceedings of the IRE **37**, 10 (1949). doi:10.1109/JRPROC.1949.232969.

[29] P. Funk, *Über Flächen mit lauter geschlossenen geodätischen Linien,* Mathematische Annalen **74**, 278 (1913). doi:10.1007/BF01456044.

[30] J. N. Lyness, *Notes on the Adaptive Simpson Quadrature Routine,* J. ACM **16**, 483–495 (1969). doi:10.1145/321526.321537.

[31] J. F. Epperson, *On the Runge Example,* The American Mathematical Monthly **94**, 329 (1987). doi:10.1080/00029890.1987.12000642.

[32] W. Palenstijn, K. Batenburg, and J. Sijbers, *Performance improvements for iterative electron tomography reconstruction using graphics processing units (GPUs),* Journal of Structural Biology **176**, 250 (2011). doi:https://doi.org/10.1016/j.jsb.2011.07.017.

[33] W. Xu, F. Xu, M. Jones, B. Keszthelyi, J. Sedat, D. Agard, and K. Mueller, *High-performance iterative electron tomography reconstruction with long-object compensation using graphics processing units (GPUs),* Journal of Structural Biology **171**, 142 (2010). doi:https://doi.org/10.1016/j.jsb.2010.03.018.

[34] F. Xu and K. Mueller, *A comparative study of popular interpolation and integration methods for use in computed tomography,* in *3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2006.*, 1252, 2006. doi:10.1109/ISBI.2006.1625152.

[35] B. Wohlberg and P. Rodriguez, *An Iteratively Reweighted Norm Algorithm for Minimization of Total Variation Functionals,* IEEE Signal Processing Letters **14**, 948 (2007). doi:10.1109/LSP.2007.906221.

[36] P. J. Huber, *Robust Estimation of a Location Parameter,* The Annals of Mathematical Statistics **35**, 73 (1964). doi:10.1214/aoms/1177703732.

[37] E. Y. Sidky, J. H. Jørgensen, and X. Pan, *Convex optimization problem prototyping for image reconstruction in computed tomography with the Chambolle–Pock algorithm,* Physics in Medicine & Biology **57**, 3065 (2012). doi:10.1088/0031-9155/57/10/3065.

[38] J. Kim, D. Pelt, M. Kagias, M. Stampanoni, K. Batenburg, and F. Marone, *Tomographic Reconstruction of the Small-Angle X-Ray Scattering Tensor with Filtered Back Projection,* Physical Review Applied **18**, (2022). Publisher Copyright: © 2022 American Physical Society. doi:10.1103/PhysRevApplied.18.014043.

[39] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk,

M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, *Array programming with NumPy,* Nature **585**, 357 (2020). doi:10.1038/s41586-020-2649-2.

[40] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,* Nature Methods **17**, 261 (2020). doi:10.1038/s41592-019-0686-2.

[41] S. K. Lam, A. Pitrou, and S. Seibert, *Numba: A llvm-based python jit compiler,* in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1, 2015.

[42] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, *scikit-image: image processing in Python,* PeerJ **2**, e453 (2014).

[43] F. Crameri, G. E. Shephard, and P. J. Heron, *The misuse of colour in science communication,* Nature Communications **11**, 5444 (2020). doi:10.1038/s41467-020-19160-7.

[44] M.-J. O. Saarinen and J.-P. Aumasson, *The BLAKE2 Cryptographic Hash and Message Authentication Code (MAC),* RFC 7693, 2015. doi:10.17487/RFC7693.

[45] M. Guizar-Sicairos, S. T. Thurman, and J. R. Fienup, *Efficient subpixel image registration algorithms,* Opt. Lett. **33**, 156 (2008). doi:10.1364/OL.33.000156.

[46] M. Odstrčil, M. Holler, J. Raabe, and M. Guizar-Sicairos, *Alignment methods for nanotomography with deep subpixel accuracy,* Opt. Express **27**, 36637 (2019). doi:10.1364/OE.27.036637.

[47] Z. Gao, M. Guizar-Sicairos, V. Lutz-Bueno, A. Schröter, M. Liebi, M. Rudin, and M. Georgiadis, *High-speed tensor tomography: iterative reconstruction tensor tomography (IRTT) algorithm,* Acta Crystallographica Section A **75**, 223 (2019). doi:10.1107/S2053273318017394.

[48] L. C. Nielsen, P. Erhart, M. Guizar-Sicairos, and M. Liebi, *Small-angle scattering tensor tomography algorithm for robust reconstruction of complex textures,* 2023. doi:10.1107/S205327332300863X.

[49] M. Guizar-Sicairos, M. Georgiadis, and M. Liebi, *Validation study of small-angle X-ray scattering tensor tomography,* Journal of Synchrotron Radiation **27**, 779 (2020). doi:10.1107/s1600577520003860.

[50] E. W. Dijkstra, *A Note on Two Problems in Connexion with Graphs.,* Numerische Mathematik **1**, 269 (1959). http://eudml.org/doc/131436.

[51] J. L. Bentley, *Multidimensional binary search trees used for associative searching,* Commun. ACM **18**, 509–517 (1975). doi:10.1145/361002.361007.

[52] D. H. Garces, W. T. Rhodes, and N. M. P. na, *Projection-slice theorem: a compact notation,* J. Opt. Soc. Am. A **28**, 766 (2011). doi:10.1364/JOSAA.28.000766.

[53] P. Trampert, W. Wang, D. Chen, R. B. Ravelli, T. Dahmen, P. J. Peters, C. Kübel, and P. Slusallek, *Exemplar-based inpainting as a solution to the missing wedge problem in elec-*

tron tomography, Ultramicroscopy **191**, 1 (2018). doi:https://doi.org/10.1016/j.ultramic.2018.04.001.

[54] J. P. Ahrens, B. Geveci, and C. C. Law, *ParaView: An End-User Tool for Large-Data Visualization,* in *The Visualization Handbook*, 2005. https://api.semanticscholar.org/CorpusID:56558637.

[55] Pliny the Elder, J. Bostock, H. T. Riley, and K. F. T. Mayhoff, *The natural history,* 2006. http://www.perseus.tufts.edu/cgi-bin/ptext?lookup=Plin.+Nat.+toc.

[56] L. De Caro, A. Terzi, L. Fusaro, D. Altamura, F. Boccafoschi, O. Bunk, and C. Giannini, *Time scale of glycation in collagen of bovine pericardium-derived bio-tissues,* IUCrJ **8**, 1024 (2021). doi:10.1107/S2052252521010344.

[57] I. Zizak, P. Roschger, O. Paris, B. Misof, A. Berzlanovich, S. Bernstorff, H. Amenitsch, K. Klaushofer, and P. Fratzl, *Characteristics of mineral particles in the human bone/cartilage interface,* Journal of Structural Biology **141**, 208 (2003). doi:https://doi.org/10.1016/S1047-8477(02)00635-4.

[58] B. Weinhausen, J.-F. Nolting, C. Olendrowitz, J. Langfahl-Klabes, M. Reynolds, T. Salditt, and S. Köster, *X-ray nano-diffraction on cytoskeletal networks,* New Journal of Physics **14**, 085013 (2012). doi:10.1088/1367-2630/14/8/085013.

[59] P. Fratzl, N. Fratzl-Zelman, K. Klaushofer, G. Vogl, and K. Koller, *Nucleation and growth of mineral crystals in bone studied by small-angle X-ray scattering,* Calcified Tissue International **48**, 407 (1991).

[60] M. Liebi, V. Lutz-Bueno, M. Guizar-Sicairos, B. M. Schönbauer, J. Eichler, E. Martinelli, J. F. Löffler, A. Weinberg, H. Lichtenegger, and T. A. Grünewald, *3D nanoscale analysis of bone healing around degrading Mg implants evaluated by X-ray scattering tensor tomography,* Acta Biomaterialia **134**, 804 (2021). doi:https://doi.org/10.1016/j.actbio.2021.07.060.

[61] E. A. Casanova, A. Rodriguez-Palomo, L. Stähli, K. Arnke, O. Gröninger, M. Generali, Y. Neldner, S. Tiziani, A. P. Dominguez, M. Guizar-Sicairos, Z. Gao, C. Appel, L. C. Nielsen, M. Georgiadis, F. E. Weber, W. Stark, H.-C. Pape, P. Cinelli, and M. Liebi, *SAXS imaging reveals optimized osseointegration properties of bioengineered oriented 3D-PLGA/aCaP scaffolds in a critical size bone defect model,* Biomaterials **294**, 121989 (2023). doi:https://doi.org/10.1016/j.biomaterials.2022.121989.

[62] H. D. Mertens and D. I. Svergun, *Structural characterization of proteins and complexes using small-angle X-ray solution scattering,* Journal of Structural Biology **172**, 128 (2010). New Trends in Protein Expression. doi:https://doi.org/10.1016/j.jsb.2010.06.012.